J. Symbolic Computation (1992) **13**, 353–394

An Algorithm for Solving Parametric Linear Systems†

WILLIAM Y. SIT

Department of Mathematics, The City College of New York, New York, NY 10031, USA
IBM Research, P. O .Box 218, Yorktown Heights, NY 10598, USA

(Received 17 April 1988)

We present a theoretical foundation for studying parametric systems of linear equations and prove an efficient algorithm for identifying all parametric values (including degenerate cases) for which the system is consistent. The algorithm gives a small set of regimes where for each regime, the solutions of the specialized systems may be given uniformly. For homogeneous linear systems, or for systems where the right hand side is arbitrary, this small set is irredundant. We discuss in detail practical issues concerning implementations, with particular emphasis on simplification of results. Examples are given based on a close implementation of the algorithm in SCRATCHPAD II. We also give a complexity analysis of the Gaussian elimination method and compare that with our algorithm.

1. Introduction

Consider the following problem: given a parametric system of linear equations (PSLE) over a computable coefficient domain R, with parameters $x = (x_1, \ldots, x_m)$, determine all choices of parametric values $\alpha = (\alpha_1, \ldots, \alpha_m)$ for which the system (which may become degenerate) is solvable, and for each such choice, solve the linear system. Here α_i lies in some computable extension field U of the quotient field F of R; for example, U may be some finite algebraic extension of $\mathbb{Q}$ when $R = \mathbb{Z}$. For obvious reasons, we would prefer the algorithm to give the solutions in the most generic form, and the set of α for which a generic solution is valid should be described as simply as possible. In this paper, we develop some theoretical results for PSLE and obtain an efficient algorithm which computes a small but complete list of conditions on the parameters under which the linear system is consistent, and solves the system for each regime uniformly. Basically, the algorithm, using computation involving only determinants, identifies a small number of regimes, and reduces the problem to a corresponding number of linear systems over the polynomial ring $R[x]$, which we then solve in $F(x)$. The proof that these regimes are sufficient is elementary and our method of selecting them is based on radical ideal membership testing. Our actual implementation uses Gröbner bases and factorization, and attempts to present the regimes and solutions in the "simplest" form.

† Portions of this paper were presented at the International Symposium on Symbolic and Algebraic Computation, Bonn, Germany, July 15–17, 1991, and appeared in its Proceedings, an ACM publication.

0747-7171/92/040353+42 $03.00/0

© 1992 Academic Press Limited

While our implementation guarantees neither minimality nor irredundancy of the regimes, both the theory and technique would, we believe, lay the foundation for future improvement.

This problem seems not to have been addressed in the literature. Non-parametric linear systems over the polynomial ring $\mathbb{C}[z,w]$ were studied by Guiver (1985), where under favorable conditions on the coefficient matrix, he derived sufficient conditions for the system to have a solution over $\mathbb{C}[z,w]$. Buchberger (1987) gave an example in robotics illustrating the solution of a (non-linear) parametric system, but the system was solved generically, without enumerating conditions on the parameters under which the system is consistent. In his approach, the parameters are treated as indeterminates, rather than as unspecified elements in an extension of R. Thus all non-zero polynomials in the parameters remain non-zero as polynomials in Buchberger's computation, while in our approach, we consider the possibility that they may become zero when the parameters are suitably specialized (the so called degenerate cases). For linear systems, his method is basically equivalent to solving a linear system over a polynomial ring by Gaussian elimination. Most recently a theoretical approach on parametric algebraic systems is given by Weispfenning (1990) based on the concept of a comprehensive Gröbner basis. His results are far too general for PSLE, and when specialized to linear systems, seem to be equivalent to the Gaussian elimination approach, which we shall discuss in §§2 and 9.

Our interest in the PSLE problem will be mainly its application to a more difficult problem: finding first integrals for parametric first order autonomous systems of ordinary differential equations, where the derivative of each unknown function is given as a multinomial expression in the unknown functions. Goldman (1987) gave a partial algorithm to solve the non-parametric case, and Sit (1988) outlined a complete and simplified algorithm. A more detailed exposition of the first integral algorithm will be forthcoming. Besides this important application, PSLEs occur naturally in many algorithms based on the method of undetermined coefficients and in finding the non-trivial equilibrium points of a system of first order ordinary differential equations such as a Lotka-Volterra system. Equilibrium points of other dynamical systems sometimes can also be found if they can be suitably transformed. For example, for biochemical systems derived with the Power-Law Formalism (see Savageau *et al.*, 1987a, b), it is possible to use our algorithm to find an explicit steady-state solution. Roughly speaking, these systems have the property that each derivative of a dependent (aggregate) variable is expressed as the difference of two multinomials, with parametric exponents (representing the kinetic orders) and coefficients (representing the rate constants). Since steady-state solution is obtained by setting each derivative to zero, the resulting algebraic system can be transformed into a parametric system of linear equations in the logarithms of the dependent variables.

This paper is organized into ten sections. In the next section, we study some simple examples and use them to illustrate the subtleties of solving a PSLE, especially by elimination schemes. Section 3 reviews some basic terminologies from classical algebraic geometry and develops an abstract setting for parametric linear equations. The

basic theory is exposed in §4. In the next two sections, we describe different versions of the algorithm and prove their correctness. We discuss implementation issues in §7, where we pay particular attention to the simplification of results using Gröbner bases techniques. We show examples from our SCRATCHPAD II (IBM) implementation in §8. Then we return to an analysis of the worst case complexity of the Gaussian elimination and compare that with our algorithm. This complexity, in a sense, measures the number of distinct ways the Gaussian elimination may be executed when applied to all possible linear systems. In the last section, we conclude with some directions for further research.

2. Gaussian Elimination

In this section, we discuss the problems in applying the usual Gaussian elimination method to a PSLE. Without going into details, we shall reveal by examples some of the inefficiencies. These examples also serve to make later sections easier to comprehend.

As everyone knows, Gaussian elimination depends on elementary row operations and the main step is pivoting, when a row is divided by a leading non-zero entry on the row. For parametric linear systems, independent of the pivoting rule to select the row, at each such step when the pivot is a non-zero, non-constant polynomial or rational function $g(x)$ in x, we must branch and consider the two cases: $g(\alpha) = 0$ or $g(\alpha) \neq 0$, where α is an actual parameter; we also have to keep track of all branches to allow backtracking and back-substitution (when a branch leads to inconsistency, or when consistency is found). This approach leads however to far too many branches (thus too many algebraic systems defining the parametric values, as well as too many back-substitutions) than are really necessary. Example 2.2 shows this for a generic 2×2 system. In §9, we shall derive the exact number of distinct paths in the generic case, and show that our algorithm produces a lot less cases. Indeed, with Gaussian elimination, the sets of α satisfying the conditions specified by distinct paths leading to either inconsistency or consistency are mutually disjoint. Many of these sets may be empty (that is, the algebraic conditions specified cannot be satisfied), and in paths leading to consistency, it is often the case that several of these sets may be merged so that a single generic solution works for all α in these sets (see Example 2.1 below). The problem of how to perform such a merge seems to be a difficult one, since in general, there are many ways to express the same solution because of the algebraic conditions on α, not to mention the non-uniqueness of a basis for the associated homogeneous system.

Gaussian elimination usually requires rational arithmetic in the coefficients, even if the given coefficients of the PSLE are polynomial in the parameters. In applying fraction free versions such as Gauss-Bareiss reduction (Bareiss, 1968), one must be

careful to keep track of all multipliers $g(x)$ and specify that $g(\alpha) \neq 0$; for otherwise, extraneous solutions are introduced for those α satisfying $g(\alpha) = 0$. Thus the comments in the preceding paragraph still apply. In contrast, the algorithm we shall present involves no pivoting, and requires mostly polynomial computations. It is modular, conceptually simple, and inherently parallel.

EXAMPLE 2.1. We consider the homogeneous PSLE L below with parameters $x = (a, b)$ and unknowns $z = (z_1, z_2, z_3)$:

$$\begin{bmatrix} -a+b & a & a^2-1 \\ b & a^2+1 & a^3 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

This system is consistent for arbitrary a, b. Its solution space is one dimensional, and a basis may be given by $z = (1, -\delta_2, \delta_1)$, where

$$\delta_1 = (a^2 - a + 1)b - a - a^3,$$
$$\delta_2 = (a^3 - a^2 + 1)b - a^4,$$

are 2×2 determinants. Our algorithm computes these, as well as $\delta_3 = 1$ and gives this single regime. Using Gauss-Bareiss reduction produces the following PSLE:

$$\begin{bmatrix} -a+b & a & a^2-1 \\ 0 & \delta_1 & \delta_2 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

We have multiplied the second equation of the system by $-a+b$ and introduced the extraneous basis $z = (1, 0, 0)$ when $a = b$ and $a \neq 0$. Indeed, the solution space given by this modified system is 2-dimensional under these conditions. If we apply care, and specify $-a+b \neq 0$ when we perform the multiplication during the Gauss-Bareiss reduction, then as far as branching is concerned, we are led to the same number of cases as with Gaussian elimination. The reader can verify (with some help from your favorite computer algebra system) that this method leads to 9 distinct paths: 5 yielding empty regimes and the remaining 4, disjoint non-empty regimes. The 4 respective bases for the solution spaces are

(1) $a \neq b$, $\delta_1 \neq 0$, $z = (1/\delta_1, -\delta_2/\delta_1, 1)$;
(2) $a \neq b$, $\delta_1 = 0$, $\delta_2 \neq 0$, $z = (a/(a-b), 1, 0)$;
(3) $a = b$, $a \neq 0$, $b \neq 0$, $z = (1/ab, (1-a^2)/a, 1)$; and
(4) $a = b$, $a = 0$, $b = 0$, $a^2 + 1 \neq 0$, $a^2 - 1 \neq 0$, $z = (1, 0, 0)$.

In deriving these regimes and bases, we have not simplified any intermediate expressions, or make substitutions, but rather have given them the way they come up during the branching process. The main challenge (theoretically) with a branch and pivot scheme is to find an algorithm to merge these 4 regimes into the single one given by our algorithm. □

EXAMPLE 2.2. Consider a 2×2 generic system L with $x = (a, b, c, d, u, v)$:

$$L: \begin{bmatrix} a & b \\ b & d \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} u \\ v \end{bmatrix}.$$

Applying Gaussian elimination, it is easy to verify that the procedure will lead to 13 branches (with mutually disjoint regimes), 6 of these ending with an inconsistent system. On the other hand, our method requires solving merely 6 systems, and only one of these involves 2 linear equations (see Example 8.1). □

EXAMPLE 2.3. We mentioned in Example 2.1 that we did not perform simplification while using Gaussian elimination. To simplify, trivial as it seems in these examples, will require either some heuristics or computations modulo a polynomial ideal in a general algorithm. Even in the case of a simple substitution like $x = 0$, one must first save the environment before the substitution in order to allow for back-tracking, and then check that specifying $x = 0$ is consistent with earlier specifications. Here is a simple example that illustrates the problems. Let x be a single indeterminate. The parametric system is:

$$\begin{bmatrix} x & x^2 \\ x^2 + 1 & x^3 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}. \quad \square$$

3. Algebraic Preliminaries and Problem Formulation

We begin by setting up some notations and recall a few basic concepts from algebraic geometry. Readers should refer to Zariski-Samuel (1958), or Lang (1964) for terms and results that are not reviewed here.

Let F be a field, with characteristic 0. Let U be a (universal) extension of F. (To carry out actual computations, F will be a computable field and U a computable extension. In practice, F will be the field $\mathbb{Q}$ of rational numbers, and one can take U to be some finite but unspecified extension of $\mathbb{Q}$. Readers unfamiliar with universal extensions may substitute U with these concrete extensions of $\mathbb{Q}$.) Let $m \in \mathbb{N}$ and let $x = (x_1, \ldots, x_m)$ be m indeterminates over U. A subset of U^m is an algebraic set defined over F (or an F-closed subset) if it is the set of common zeros in the affine space U^m of a (finite) set of polynomials in $F[x]$. The complements of F-closed subsets are called F-open subsets, and they form the basis of a topology (the Zariski F-topology) on U^m. Henceforth, the terms "open" and "closed" will refer to this topology. If h is a family, or a vector, or a subset of elements in $F[x]$, we shall denote the set of common zeros of h in U^m by $V^m(h)$ and its complement by $\overline{V}^m(h)$. When m is clear from the context, we shall often simply use the notation V and $\overline{V}$.

Given a (finite) set h of polynomials in $F[x]$, let Ideal(h) be the ideal generated by h and let Rad(h) be the radical ideal generated by h. (A more precise notation would be $\text{Ideal}_{F[x]}(h)$ and $\text{Rad}_{F[x]}(h)$, but we shall often omit mentioning the polynomial ring when it is clear from the context.) Then $V(h) = V(\text{Ideal}(h)) = V(\text{Rad}(h))$. A subset is *quasi-algebraic* (or *locally-closed*) if it is the intersection of an algebraic set and an open set. For example, the set of points $(x,0)$ on the xy-plane such that $x \neq 0$ is quasi-algebraic since it is $V(y) \cap \overline{V}(x)$.

Let $\alpha \in U^m$, and $f(x) \in F(x)$. We say $f(x)$ is *defined at* α if $f(x)$ can be written in the form $p(x)/q(x)$ where $p(x), q(x) \in F[x]$ and $q(\alpha) \neq 0$. Given $f(x)$, the set of α for which $f(x)$ is defined at α is an open set called the *domain* of the rational function $f(x)$. If $\mathbf{f} = (f_1(x), \ldots, f_k(x))$ is a vector of rational functions, then the domain of $\mathbf{f}$ is the intersection of the domains of all f_i. Finally, if S is a subset of U^m we say $\mathbf{f}$ is *defined on* S if it is defined at every $\alpha \in S$. Two rational functions f, g are *equivalent on* S if they are both defined on S and $f - g$ vanishes on S.

We now give a precise formulation of the problem. Because we have in mind several specific applications (for first integrals of autonomous systems), our treatment will be more general than is needed for normal applications. Basically, we would like to treat the parameters that appear only on the right-hand side separately from those appearing on the left (possibly both) side(s). This has the advantage of having less number of indeterminates in certain steps of the algorithm, an important consideration in any computer implementation. Moreover, we shall deal exclusively with PSLE which involve the parameters only polynomially. Most PSLE of interest can be transformed into this category.

Let R be an integral domain, F be its quotient field, and let U be a universal extension of F. Let $x = (x_1, \ldots, x_m)$, $w = (w_1, \ldots, w_r)$, and $z = (z_1, \ldots, z_n)$ be three independent families of indeterminates over U. A *parametric* system of linear equations (PSLE) over R with parameters (x, w) is a linear system

$$L: \quad C(x)z = A(x,w), \tag{1}$$

given by the unknown column vector z, an $r \times n$ coefficient matrix $C = C(x) = [C_{kl}(x)]$ and a right-hand side column vector $A = A(x,w) = (A_1(x,w), \ldots, A_r(x,w))$, where $C_{kl}(x) \in R[x]$ and $A_j(x,w) \in R[x,w]$. We call the affine space U^m the *parameter space* of L and denote this by X. We call the affine space U^{m+r} the *extended parameter space* of L and denote it by X^r. Let L^0 denote the homogeneous PSLE associated with L. For any pair $(\alpha, \beta) \in X^r$, let $L_{(\alpha,\beta)}$ denote the linear system $C(\alpha)z = A(\alpha, \beta)$.

Next, we introduce the concepts of solution functions and regimes. A *solution function* of L is a pair (S, Z), where S is a non-empty subset of X^r and Z is an $n \times (v+1)$ matrix of rational functions in $F(x,w)$ with columns $Z_0, Z_1, \ldots, Z_v$, for some v, $0 \leq v \leq n$, such that for all $(\alpha, \beta) \in S$, we have (a) the entries of Z are defined at (α, β), (b) $Z_0(\alpha, \beta)$ is a particular solution of $L_{(\alpha,\beta)}$, and (c) $(Z_1(\alpha, \beta), \ldots, Z_v(\alpha, \beta))$ is a basis of the homogeneous system $L^0_{(\alpha,\beta)}$. We say a non-empty subset S of X^r is a *regime* of L if there exists a Z as above such that (S, Z) is a solution function of L. By abuse use of language, we often call Z a solution function on S. We denote the domain of Z by

dom(Z). The largest subset T of X^r such that (T, Z) is a solution function of L will be denoted by $S(Z)$. Of course, $S \subseteq S(Z)$.

REMARK. Note that in (c), for a fixed (α, β), the homogeneous system, and hence also the basis, depend only on α. We could have defined the entries of $Z_1, \ldots, Z_v$ to be in $F(x)$, and the entries of Z_0 to be in $F(x)[w]$. However, as we shall see, α and β are not necessarily independent. Thus $Z_1, \ldots, Z_v$ may involve w. We use the more general definition and notation so as not to restrict *a priori* our freedom in *representing* solution functions. In this sense, the notation $L^0_{(\alpha,\beta)}$ reminds us that *on* S the homogeneous system L^0 may be equivalent to one involving both x and w. □

Let S be a regime of L and let Z be a solution function on S. Let $(\alpha, \beta) \in S$. A basis of the vector space of solutions of the homogeneous system $L^0_{(\alpha,\beta)}$ has v elements. Hence $v \leq n$ and $\operatorname{rank}(C(\alpha)) = n - v$, which on the one hand, is independent of (α, β), and on the other, is independent of Z. We call $n - v$ the *C-rank*, or simply, the *rank* of S and denote it by $c(S)$. So $0 \leq c(S) \leq \min(r, n)$. Since the pair $(S(Z), Z)$ is also a solution function we have $c(S) = c(S(Z))$. More generally, we have:

LEMMA 3.1. *Let S^1 and S^2 be two regimes of L. If $c(S^1) \neq c(S^2)$ then S^1 and S^2 are disjoint.* □

Let $\Lambda(L)$ be the set of all points $(\alpha, \beta) \in X^r$ for which $L_{(\alpha,\beta)}$ is consistent. If S is a regime of L then clearly $S \subseteq \Lambda(L)$. Let $\mathbf{S} = \{S^1, \ldots, S^t\}$ be a family of regimes of L. We shall say the family $\mathbf{S}$ *covers* L (or is a cover for L) if $\Lambda(L) = \bigcup S^i$. We say $\mathbf{S}$ is an *irredundant cover* if $\mathbf{S}$ covers L and no proper subfamily of $\mathbf{S}$ covers L. Finally, we say $\mathbf{S}$ is a *minimum cover* if $\mathbf{S}$ covers L and there is no cover $\mathbf{T} = \{T^1, \ldots, T^t\}$ of L with $t < s$. A minimum cover is always irredundant. For a given PSLE L, we are interested in an efficient algorithm to compute a minimum cover $\mathbf{S}$ of $\Lambda(L)$. In other words, we want to express $\Lambda(L)$ as the union of a minimum number of regimes S^i, where on each regime we can solve L uniformly by some solution function Z^i.

In the remainder of this section, we illustrate with some examples.

EXAMPLE 3.2. For the system L of Example 2.1, a minimum cover for L is $\{(X, Z)\}$, where $X = U^2 = \Lambda(L)$ and

$$Z = \begin{bmatrix} 0 & 1 \\ 0 & -\delta_2 \\ 0 & \delta_1 \end{bmatrix}. \quad \square$$

EXAMPLE 3.3. Let $x = (a, b)$ be the parameters, and consider the 3×3 linear system L:

$$\begin{aligned} z_1 + az_2 + bz_3 &= 1, \\ bz_1 + z_2 + az_3 &= 1, \\ az_1 + bz_2 + z_3 &= 1. \end{aligned}$$

In this example, $\tau = 0$. Let $S = \overline{V}(\delta(x))$ where

$$\delta(x) = (a+b+1)(a^2 - ab - a + b^2 - b + 1)$$

is the 3×3 determinant of the coefficient matrix. Let Z be the matrix with a single column vector

$$Z_0 = (\frac{1}{b+a+1}, \frac{1}{b+a+1}, \frac{1}{b+a+1}).$$

Then $c(S) = 3$, $(1,0) \in S$ and (S,Z) is a solution function for L and indeed, $S = S(Z)$. We have $\mathrm{dom}(Z) = \overline{V}(b+a+1) = \Lambda(L)$. We note that $(1,1) \in \mathrm{dom}(Z)$ and $Z_0(1,1)$ *is* a particular solution of $L_{(1,1)}$, but $(1,1) \notin S(Z)$. Thus, in general, $S(Z) \neq \mathrm{dom}(Z) \cap \Lambda(L)$. □

EXAMPLE 3.4. Consider L as in Example 3.3. Let

$$S = V(a^2 - ab - a + b^2 - b + 1) \cap \overline{V}(a - b^2)$$

and let Z be the matrix with column vectors Z_0, Z_1 where,

$$Z_0 = \left(\frac{b-a}{b^2-a}, 0, \frac{b-1}{b^2-a}\right), \quad Z_1 = \left(\frac{-b+a^2}{b^2-a}, 1, \frac{-ab+1}{b^2-a}\right).$$

Then $c(S) = 2$, $(0, \omega) \in S$, where ω is a primitive cube root of -1, and (S, Z) is a solution function of L. Let Y be the matrix with column vectors Y_0, Y_1 where,

$$Y_0 = \left(\frac{b-1}{ab-1}, 0, \frac{a-1}{ab-1}\right), \quad Y_1 = \left(\frac{-b^2+a}{ab-1}, 1, \frac{b-a^2}{ab-1}\right).$$

It can be easily checked that the function $ab - 1$ is never zero on S and that Y is also a solution function on S. In *this* example, the two solution functions are equivalent (that is, each corresponding rational function entries are equivalent) on S. Of course, solution functions for the same regime need not be equivalent in general. □

EXAMPLE 3.5. Again consider the system L of Example 3.3. Let S^1 be the set consisting of the single point $(1,1)$. Then clearly S^1 is a regime of L, with a particular solution $(1, 0, 0)$ and a basis $\{(-1, 0, 1), (-1, 1, 0)\}$. Let S^2 be the regime in Example 3.4, and let S^3 be the regime in Example 3.3. Since $\Lambda(L) = \overline{V}(b+a+1)$, the family $\{S^1, S^2, S^3\}$ is a minimum cover of L (see also Corollary 4.3). □

REMARK. For the linear system in the examples above, if we are only interested in a particular solution, then we only need one solution function on $\Lambda(L)$, namely, Z in Example 3.3.

4. Special Solution Functions

Our first result (Theorem 4.1) explicitly constructs a finite cover for L. The proof is based on the simple consistency condition of a linear system. Let $(\alpha, \beta) \in \Lambda(L)$. Then $\text{rank}(C(\alpha)) = \text{rank}(C(\alpha), A(\alpha, \beta))$. If this rank is c, the consistency condition is equivalent to the requirement that all $(c+1) \times (c+1)$ subdeterminants of both the coefficient matrix and the augmented matrix vanish at (α, β), while some $c \times c$ subdeterminant of the coefficient matrix does not. Thus $\Lambda(L)$ is the union of a finite number of quasi-algebraic subsets of X^r. On each of these quasi-algebraic subsets, we can obtain a solution function explicitly. The theorem thus provides a crude algorithm for solving PSLE. Later, using these explicit formulæ on the regimes, we develop methods to merge different regimes and reduce their number to arrive at a more efficient algorithm. We need a few more definitions and notations.

Let $c \in \mathbb{N}$, $0 \leq c \leq \min(r, n) + 1$, and let Δ_c be a complete set of non-zero determinants of $c \times c$ submatrices of $C(x)$, where $\Delta_0 = \{1\}$ and $\Delta_{\min(r,n)+1} = \phi$ by convention. Obviously, $\Delta_c \subseteq F[x]$ for every c. Let π be the projection of X^r onto X. For any solution function (S, Z), and any $(\alpha, \beta) \in S$, the determinant of every $(c(S)+1) \times (c(S)+1)$ submatrix of $C(\alpha)$ must be zero. Thus $\pi(S)$ is a subset of the quasi-algebraic set $V(\Delta_{c(S)+1}) \cap \pi(\text{dom}(Z))$; in particular, the latter set is non-empty, since S is non-empty by definition.

Let $S \subseteq X^r$ and $\alpha \in \pi(S)$. We define the *fiber* of S over α to be the set $S^\alpha = \{\beta \in U^r \mid (\alpha, \beta) \in S\}$. The set S is said to have *generic* F-fibers if there exist polynomials $h_1(x,w), \ldots, h_s(x,w) \in F[x,w]$ such that $S^\alpha = V^r(h_1(\alpha, w), \ldots, h_s(\alpha, w))$ for each $\alpha \in \pi(S)$.

THEOREM 4.1. *Let L be a PSLE as given by* (1). *Then we can construct k regimes $S^1, \ldots, S^k$ covering L, and for each i, $1 \leq i \leq k$, a solution function Z^i on S^i. Moreover, we have*

$$k \leq \binom{n+r}{r},$$

and each S^i is a quasi-algebraic subset of X^r having generic F-fibers.

PROOF. Let $c \in \mathbb{N}$, $0 \leq c \leq \min(r, n)$. Fix a c-subset a of $\{1, \ldots, r\}$ and let $\bar{a}$ be the complement of a. Similarly fix a c-subset b of $\{1, \ldots, n\}$ and let $\bar{b}$ be the complement of b. Let C^{ab} be the $c \times c$ submatrix of $C(x)$ consisting of entries C_{kl} with $k \in a$ and $l \in b$. Let z^b be the subvector $(z_l)_{l \in b}$ of z. Let A^a be the subvector $(A_k)_{k \in a}$ of A. In what follows, other submatrices and subvectors will be similarly notated. Let $\delta(x) = \delta_{ab}(x)$ be the determinant of $C^{ab}(x)$ (if $c = 0$, let $\delta(x) = 1$). Suppose $\delta(x) \neq 0$. We are going to construct an $n \times (v+1)$ matrix Z where $v = n - c$, with entries in $F(x)[w]$. Without loss of generality, we shall suppose that C^{ab} is given by the first c columns and c rows of the matrix C. Thus we can partition the matrix C and the vectors z and A as follows:

$$C=\begin{bmatrix} C^{ab} & C^{a\bar{b}} \\ C^{\bar{a}b} & C^{\bar{a}\bar{b}} \end{bmatrix}, \quad z=\begin{bmatrix} z^{b} \\ z^{\bar{b}} \end{bmatrix}, \text{ and } A=\begin{bmatrix} A^{a} \\ A^{\bar{a}} \end{bmatrix}. \tag{2}$$

The linear system L may be written as:

$$C^{ab}(x)z^{b}+C^{a\bar{b}}(x)z^{\bar{b}}=A^{a}(x,w), \tag{3(x, w)}$$

$$C^{\bar{a}b}(x)z^{b}+C^{\bar{a}\bar{b}}(x)z^{\bar{b}}=A^{\bar{a}}(x,w). \tag{4(x, w)}$$

Now, let $K(x)=K_{ab}(x)$ be the inverse of $C^{ab}(x)$ if $c\neq 0$ and let $K(x)=1$ otherwise. Define the matrix $Z=Z_{ab}(x,w)$ by

$$Z_{ab}(x,w)=\begin{bmatrix} K_{ab}(x)A^{a}(x,w) & -K_{ab}(x)C^{a\bar{b}}(x) \\ 0 & J \end{bmatrix}, \tag{5}$$

where J is a $\nu\times\nu$ identity matrix. Note that for general a and b, a suitable permutation of the rows of right hand side in (5) is implicit in the definition of Z_{ab}; note also, when $c=0$, $Z_{ab}=[0\ J]$. Let $Z_0, Z_1, \ldots, Z_\nu$ be the columns of Z. Clearly, $\overline{V}(\delta(x))\subseteq\mathrm{dom}(Z)$. Let $\mu=r-c$ and let $h(x,w)=h_{ab}(x,w)$ be defined by

$$\begin{aligned} h_{ab}(x,w)&=0 && \text{if } \mu=0, \text{ and}\\ h_{ab}(x,w)&=\delta(x)\big(C^{\bar{a}b}(x)Z_0{}^{b}(x,w)+C^{\bar{a}\bar{b}}(x)Z_0{}^{\bar{b}}(x,w)-A^{\bar{a}}(x,w)\big)\\ &=\delta(x)\big(C^{\bar{a}b}(x)Z_0{}^{b}(x,w)-A^{\bar{a}}(x,w)\big)\\ &=\delta_{ab}(x)\big(C^{\bar{a}b}(x)K_{ab}(x)A^{a}(x,w)-A^{\bar{a}}(x,w)\big) && \text{if } \mu>0. \end{aligned} \tag{6}$$

Note that $h(x,w)\in F[x,w]^{\mu}$ and $h(x,w)=-A(x,w)$ when $c=0$. Finally, we let

$$S=S_{ab}=\overline{V}^{m+\tau}(\delta(x))\cap V^{m+\tau}(\Delta_{c+1}(x),h(x,w)). \tag{7}$$

Then S is a quasi-algebraic set defined over F. We claim that (S,Z) is a solution function, *provided that* $S\neq\phi$. Let $(\alpha,\beta)\in S$. Then the determinant of every $(c+1)\times(c+1)$ submatrix of $C(\alpha)$ must be zero (if $c=\min(r,n)$, this condition is vacuous). Moreover, since $\delta(\alpha)\neq 0$, the matrix $C^{ab}(\alpha)$ is invertible and has inverse $K(\alpha)$. Thus $\mathrm{rank}(C(\alpha))=c$ and the system $L_{(\alpha,\beta)}$ can be rewritten as $(3(\alpha,\beta))$ and $(4(\alpha,\beta))$. Using $(3(\alpha,\beta))$, we can solve uniquely for z^{a}, in terms of the remaining unknowns $z^{\bar{b}}$, which can be arbitrary. Thus, $Z_1(\alpha,\beta),\ldots,Z_\nu(\alpha,\beta)$ are linearly independent solutions of the homogeneous system corresponding to $(3(\alpha,\beta))$. Since $\mathrm{rank}(C(\alpha))=c$, they are also solutions to the homogeneous system corresponding to $(4(\alpha,\beta))$. Now, $Z_0(\alpha,\beta)$ is a particular solution of $(3(\alpha,\beta))$. Since $h(\alpha,\beta)=0$, it is also a solution of $(4(\alpha,\beta))$. Thus Z is a solution function on S.

It follows that for each $\alpha\in\pi(S)$, the fiber S^{α} is given by the algebraic set of zeros of $h(\alpha,w)\in F(\alpha)[w]^{\mu}$. Thus S has generic F-fibers.

To complete the proof, let $(S^1,Z^1),\ldots,(S^k,Z^k)$ be all the solution functions (S,Z) constructed as above (with non-empty S). Each such (S,Z) is determined by a choice of a non-singular (square) submatrix of $C(x)$, and there are at most λ of these, where

$$\lambda = \sum_{c=0}^{\min(r,n)} \binom{r}{c}\binom{n}{c} = \binom{n+r}{r}. \tag{8}$$

Clearly, $\bigcup_{i=1}^{k} S^i \subseteq \Lambda(L)$. Let $(\alpha, \beta) \in \Lambda(L)$. Then the ranks of the coefficient matrix $C(\alpha)$ and the augmented matrix $(C(\alpha), A(\alpha, \beta))$ are equal, say to c, $0 \leq c \leq \min(r, n)$. At least one $c \times c$ submatrix $C^{ab}(\alpha)$ of $C(\alpha)$ is nonsingular. We note that the corresponding submatrix $C^{ab}(x)$ of $C(x)$ must then also be nonsingular. Let (S, Z) be the pair constructed as above using this submatrix. Since any solution to $(3(\alpha, \beta))$ will automatically satisfy $(4(\alpha, \beta))$, we see that $(\alpha, \beta) \in S$. In particular, S is nonempty, and hence $(S, Z) = (S^i, Z^i)$ for some i. This completes the proof. □

The proof of Theorem 4.1 is constructive, and provides us with a crude algorithm for solving PSLE. We skip the description of this algorithm in this form. We shall give instead first a slightly improved version, and later a still more efficient one.

It should be noted that the bound given in Theorem 4.1 is best possible. This bound will be attained whenever the given PSLE is completely general, as in Example 2.2. In practice, however, relatively few of the entries in the coefficient matrix C involve parameters. In addition, the possible ranks of the matrices $C(\alpha)$ do not always range from 0 through $\min(r, n)$. Our first improvement takes advantage of this property. We define the *minimum rank* $\rho(C)$ of the matrix $C(x)$ to be the least rank of $C(\alpha)$ as α ranges over X. Thus $0 \leq \rho(C) \leq \min(r, n)$. Note that it is possible for $\rho(C) = 0$. An example of this is when x is a single indeterminate, and

$$C(x) = \begin{bmatrix} x & x^2 \\ x^2 + x & x^3 \end{bmatrix}.$$

On the other hand, it is quite likely that $\rho(C) > 0$, for example, when C contains some $c \times c$ nonsingular submatrix whose determinant does not involve parameters at all, in which case $\operatorname{rank}(C(\alpha)) \geq c$ for any α and hence $\rho(C) \geq c$. More generally, the proposition below holds. Recall that $\Delta_0 = \{1\}$ and that $\Delta_{\min(r,n)+1} = \phi$ by convention.

PROPOSITION 4.2. *The minimum rank* $\rho(C)$ *of* $C(x)$ *is the largest* c, $0 \leq c \leq \min(r, n)$, *such that* $V(\Delta_c) = \phi$, *or equivalently,* $1 \in \operatorname{Ideal}(\Delta_c)$.

PROOF. Let c^* denote the largest c such that $V(\Delta_c) = \phi$. For any $\alpha \in X$, let $c = \operatorname{rank}(C(\alpha))$. Suppose $c < c^*$. Then $c + 1 \leq c^*$ and $\alpha \in V(\Delta_{c+1}) \subseteq V(\Delta_{c^*}) = \phi$. This contradiction proves $\rho(C) \geq c^*$. Conversely, let $\alpha \in V(\Delta_{c^*+1})$, which is non-empty by definition of c^*. Since $X = \overline{V}(\Delta_{c^*})$, there is some $c^* \times c^*$ submatrix whose determinant does not vanish at α. Thus $\operatorname{rank}(C(\alpha)) = c^*$ and so $\rho(C) = c^*$. The equivalence is simply Hilbert's Nullstellungsatz. □

To apply Proposition 4.2 to our algorithm involves ideal membership testing, which is relatively expensive. A simpler application, but not as exact, is the following. Suppose C contains some $c \times c$ nonsingular submatrix whose determinant is a non-zero

constant, and let $\tilde{c}$ denote the largest such c. We have seen $\rho(C) \geq \tilde{c}$. Since non-zero constants are easy to detect, we can apply this in our algorithm. However, one should note that the inequality can be strict. For example, with x as a single indeterminate, let

$$C(x) = \begin{bmatrix} x & 0 & x-1 & 0 \\ 0 & x & 1 & x+1 \end{bmatrix}.$$

Then $\rho(C) = 2$ and $\tilde{c} = 1$. Of course, $\tilde{c}$ need not exist for arbitrary $C(x)$, and if $\tilde{c}$ exists, it is not necessarily true that for $c < \tilde{c}$, Δ_c contains a non-zero constant (see Example 2.1). Thus any search for a non-zero constant determinant should be done in descending order of c.

COROLLARY 4.3. *If $\mathbf{S} = \{S^1, \ldots, S^t\}$ is a family of regimes covering L such that for each c, $\rho(C) \leq c \leq \min(r, n)$, there is at most one S^j with $c(S^j) = c$, then $\mathbf{S}$ is a minimum cover of L.*

PROOF. This follows from Lemma 3.1 and the fact that $c(S) \geq \rho(C)$ for any regime S. □

Our next improvement depends crucially on the set inclusion relationships among the quasi-algebraic sets defined by (7). We begin with some definitions.

A *special set* for L is a quasi-algebraic set $S = S_{ab}$, which may be empty, corresponding to a non-singular submatrix of $C(x)$ as defined by equation (7). Note that $S_{\phi\phi} = V^{m+\tau}(\Delta_1, A(x,w))$ is a special set (case $c = 0$). The size c of the index sets a, b will be called the C-rank of S and denoted by $c(S)$. The matrix $Z = Z_{ab}$ as defined by (5) is called the *associated matrix* of S. The function $\delta(x) = \delta_{ab}(x)$ is called the *denominator* of S and the functions $h(x, w) = h_{ab}(x,w)$ defined by (6) are called the *consistency functions* of S. For a special set S, we define a quasi-algebraic set $N(S)$ by

$$N(S) = \overline{V}^m(\delta(x)) \cap V^m(\Delta_{c+1}(x)). \tag{9}$$

Clearly, by (7), $\pi(S) \subseteq N(S)$; in particular, if $N(S) = \phi$ then $S = \phi$.

A *special regime* is a special set that is non-empty (that is, one of the S^i in the proof of Theorem 4.1). For such S, $N(S)$ is non-empty. It should be emphasized that $N(S)$ may be empty for a special set. The system in Example 2.3 has just one special set S with $c(S) = 1$ and $N(S) \neq \phi$. On the other hand, the same example shows it is also possible for $N(S) \neq \phi$ but $S = \phi$. The situation is much better in the case L is sort of homogeneous.

We say the PSLE L as given in (1) is *semi-homogeneous* if $A(x,0) = 0$. A homogeneous PSLE is clearly semi-homogeneous. A PSLE with arbitrary right hand side is semi-homogeneous, for we may take $\tau = r$ and $A_j(x, w) = w_j$ for $1 \leq j \leq r$.

PROPOSITION 4.4. *With notation as in Theorem 4.1, suppose that L is semi-homogeneous. Then $\pi(S) = N(S)$ for any special set S; in particular, S is a special regime if and only if $N(S) \neq \phi$. Thus, $\pi(\Lambda(L)) = \bigcup_{i=1}^{k} N(S^i)$ is the finite union of quasi-algebraic sets.*

PROOF. For any $\alpha \in X$, and for any special set S, the algebraic system $h(\alpha, w) = 0$ has a solution in U^r, namely, the trivial solution. Hence by (7) and (9), $\alpha \in \pi(S)$ if and only if $\alpha \in N(S)$. The rest of the proposition follows from the theorem. □

PROPOSITION 4.5. *Let L be a PSLE as defined by* (1) *and let S, T be special sets for L. Let $c = c(S)$ and $c' = c(T)$.*

(a) *If $c \neq c'$, then $N(S)$ and $N(T)$ are disjoint.*
(b) *If $N(S) \subseteq N(T)$, then $S \subseteq T$; and if moreover, $N(S) \neq \phi$ then $c = c'$.*

PROOF. We may suppose $c < c'$. Now $N(S) \cap N(T)$ is easily seen to be the empty set since the denominator $\delta(x)$ of T, being a $c' \times c'$ determinant, belongs to $\text{Ideal}(\Delta_{c+1})$. For part (b), the first part is trivial if $S = \phi$. Assume $S \neq \phi$ and $(\alpha, \beta) \in S$. Then $L_{(\alpha,\beta)}$ is consistent and the hypothesis implies that $\alpha \in N(S) \subseteq N(T)$. Hence, by an argument similar to the last paragraph in the proof of Theorem 4.1, $(\alpha, \beta) \in T$. The last statement follows from (a). □

5. Algorithm for PSLE (First Version)

We now describe a first version of the algorithm. We shall assume that the following procedures are available. For simplicity, we shall assume the procedures work for all polynomial rings. The method for implementing these procedures will be discussed in a separate section.

PROCEDURE 1. Determinants($C(x)$, c). Here, $C(x)$ is an $r \times n$ matrix over a polynomial ring $R[x]$ and c is a natural number, $0 \leq c \leq \min(r, n) + 1$. This procedure returns the empty set if $c = \min(r, n) + 1$ and the set consisting of the single element 1 if $c = 0$. For $1 \leq c \leq \min(r, n)$, it returns the *set* Δ_c of all non-zero determinants of $c \times c$ submatrices of $C(x)$ *if none of these determinants are constants*; otherwise, it returns $\{\delta\}$ where δ is one such non-zero, constant determinant. We assume that any determinant returned by this routine carries with it the row index set a and the column index set b for the submatrix of which it is the determinant. In case $c = 0$ one may assign $a = b = \phi$ or use any other convenient convention.

PROCEDURE 2a. Consistency(L, a, b). Given a PSLE L as in (1), a c-subset a of $\{1, \ldots, r\}$, and a c-subset b of $\{1, \ldots, n\}$ such that C^{ab} is non-singular, this procedure returns the function h_{ab} as given by (6). Note that when $c = 0$, it returns $h_{ab} = -A(x, w)$.

PROCEDURE 2b. Solve(L, a, b). Given a PSLE L as in (1), a c-subset a of $\{1, \ldots, r\}$, and a c-subset b of $\{1, \ldots, n\}$ such that C^{ab} is non-singular, this procedure returns the matrix Z_{ab} as given by (5). Note that when $c = 0$, it returns $Z_{ab} = [0 \ J]$.

In Procedures 2a and 2b, no assumption is made that the quasi-algebraic set defined by (7) is non-empty.

PROCEDURE 3. HasSolution(h, f). Given polynomials $h = (h_1, \ldots, h_s)$ and a polynomial f with coefficients in F, this procedure returns the boolean value *TRUE* if the quasi-algebraic set $S = V(h) \cap \overline{V}(f)$ is non-empty, and the value *FALSE* otherwise.

ALGORITHM 1.

Input: A PSLE $L{:}C(x)z = A(x,w)$ as given by (1).

Output: A list of special regimes, together with their solution functions, satisfying the properties stated in Theorem 4.1.

STEP 1.
For $c := \min(r,n) + 1$ to 0 by -1 do
$D_c :=$ Determinants($C(x)$, c)
if D_c consists of a single constant, go to STEP 2
if HasSolution(D_c, 1) = *FALSE*, go to STEP 2

STEP 2.
$\rho := \max(c, 0)$
For $c := \min(r, n)$ to ρ by -1 do
For $\delta_{ab}(x) \in D_c$ do
$h_{ab}(x, w) =$ Consistency(L, a, b)
$\Phi_{ab} := D_{c+1} \bigcup \{h_{ab}(x,w)\}$
if HasSolution(Φ_{ab}, $\delta_{ab}(x)$) then
$Z_{ab}(x,w)$: = Solve(L, a, b)
output Φ_{ab}, $\delta_{ab}(x)$, Z_{ab}.

PROOF OF CORRECTNESS. By Theorem 4.1, we need only consider special regimes. Let $0 \leq c \leq \min(r, n)$. For any c-subset a of $\{1, \ldots, r\}$ and b of $\{1, \ldots, n\}$, if S_{ab} is a special regime then $N(S_{ab}) \neq \phi$, and $\delta_{ab}(x) \neq 0$. Duplicate determinants may be eliminated since they can be dropped by Proposition 4.5(b). To justify the specifications in Procedure 1, suppose the minimum rank is reached by finding a non-zero constant $c \times c$ determinant $\delta_{ab}(x)$. Then $V^{m+\tau}(\Delta_c) = \phi$ and hence by (9) and (7), $N(S_{a'b'})$ and $S_{a'b'}$ are both empty for any $(c-1)$-subsets a' of $\{1, \ldots, r\}$ and b' of $\{1, \ldots, n\}$. Thus there is no need to compute any determinants of order less than c. Moreover, $N(S_{ab}) = V^m(\Delta_{c+1}(x)) \supseteq N(S)$ for any S with C-rank c so that by Proposition 4.5(b), $S_{ab} \supseteq S$. There is no need to consider any other special regimes with C-rank $= c$, and thus there is no need to compute any other determinants of $c \times c$ submatrices. By Proposition 4.2, we have

reached the minimum rank $\rho(C)$ when we start Step 2, and so the iteration over c in both steps may be ended. In Step 2, we simply compute h_{ab}, discard those for which $S_{ab} = V(\Phi_{ab}) \cap \overline{V}(\delta_{ab}(x))$ is empty, and compute Z_{ab} for those for which S_{ab} is not empty. □

EXAMPLE 5.1. Consider the 2×2 generic system L of Example 2.2. The 6 special regimes which irredundantly covers L, are given in Figure 1. □

Defining Equations for S	Solution Functions Z
$ad - bc \neq 0$	$\begin{bmatrix} \frac{du - bv}{ad - bc} \\ \frac{av - cu}{ad - bc} \end{bmatrix}$
$ad - bc = 0,\ a \neq 0,\ cu - av = 0$	$\begin{bmatrix} \frac{u}{a} & -\frac{b}{a} \\ 0 & 1 \end{bmatrix}$
$ad - bc = 0,\ b \neq 0,\ du - bv = 0$	$\begin{bmatrix} 0 & 1 \\ \frac{u}{b} & -\frac{a}{b} \end{bmatrix}$
$ad - bc = 0,\ c \neq 0,\ av - cu = 0$	$\begin{bmatrix} \frac{v}{c} & -\frac{d}{c} \\ 0 & 1 \end{bmatrix}$
$ad - bc = 0,\ d \neq 0,\ bv - du = 0$	$\begin{bmatrix} 0 & 1 \\ \frac{v}{d} & -\frac{c}{d} \end{bmatrix}$
$a = b = c = d = 0,\ u = v = 0$	$\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Figure 1. Regimes and Solution Functions for 2x2 system.

6. Algorithm for PSLE (Second Version)

We now investigate the more general problem of determining the set inclusion relations among the special sets. We consider special sets rather than special regimes because we have no *a priori* way of knowing when a special set is non-empty. In Algorithm 1, we have to compute the consistency functions for all special sets even though some special sets may be redundant or empty. In our improved algorithm, we shall avoid some of this. It turns out that a large portion of the set inclusion relations may be characterized by using only the algebraic relations among the determinants. We state a trivial lemma from set theory. Let D be a set. For any subset A of D, we denote its complement in D by $\overline{A}$.

LEMMA 6.1. *Let A be a subset of a set D, let $\{B_i\}$, and $\{C_j\}$ be two (possibly empty) families of subsets of D. Then*

$$A \cap (\bigcup \overline{B}_i) \subseteq A \cap (\bigcup \overline{C}_j) \Leftrightarrow A \cap (\bigcap B_i) \supseteq A \cap (\bigcap C_j). \quad \square$$

PROPOSITION 6.2. *Let $\{H^i\} = \{V(\Theta) \cap \overline{V}(p_i)\}$ and $\{K^j\} = \{V(\Theta) \cap \overline{V}(q_j)\}$ be two families of quasi-algebraic sets, where $\Theta \subseteq F[x]$ and $p_i, q_j \in F[x]$. Then*

$$\bigcup H^i \subseteq \bigcup K^j \Leftrightarrow \{p_i\} \subseteq \mathrm{Rad}(\Theta, \{q_j\}).$$

PROOF. We have, by applying Lemma 6.1,

$$\begin{aligned}
&\bigcup H^i \subseteq \bigcup K^j \\
\Leftrightarrow\ &\bigcup (V(\Theta) \cap \overline{V}(p_i)) \subseteq \bigcup (V(\Theta) \cap \overline{V}(q_j)) \\
\Leftrightarrow\ &V(\Theta) \cap (\bigcup \overline{V}(p_i)) \subseteq V(\Theta) \cap (\bigcup \overline{V}(q_j)) \\
\Leftrightarrow\ &V(\Theta) \cap (\bigcap V(p_i)) \supseteq V(\Theta) \cap (\bigcap V(q_j)) \\
\Leftrightarrow\ &V(\Theta, \{p_i\}) \supseteq V(\Theta, \{q_j\}) \\
\Leftrightarrow\ &\mathrm{Rad}(\Theta, \{p_i\}) \subseteq \mathrm{Rad}(\Theta, \{q_j\}) \\
\Leftrightarrow\ &\{p_i\} \subseteq \mathrm{Rad}(\Theta, \{q_j\}). \quad \square
\end{aligned}$$

COROLLARY 6.3. *Let $\{T^i\}$ and $\{S^j\}$ be two families of special sets of (1) all having the same C-rank c. Let $\{\gamma_i(x)\}$ and $\{\delta_j(x)\}$ be the corresponding family of denominators. Then*

$$\bigcup N(T^i) \subseteq \bigcup N(S^j) \Leftrightarrow \{\gamma_i(x)\} \subseteq \mathrm{Rad}(\Delta_{c+1}, \{\delta_j(x)\}).$$

Moreover, when this condition is satisfied, we have $\bigcup T^i \subseteq \bigcup S^j$.

PROOF. The first statement follows from the proposition by taking $H^i = N(T^i)$, $K^j = N(S^j)$, $\Theta = \Delta_{c+1}$, $p_i = \gamma_i$, and $q_j = \delta_j$. To prove the second statement, let $(\alpha, \beta) \in T^i$. Then $L_{(\alpha,\beta)}$ is consistent and $\alpha \in N(T^i)$. Now, for some j, $\alpha \in N(S^j)$ and hence $(\alpha, \beta) \in S^j$. $\square$

THEOREM 6.4. *Let L be a PSLE as given by (1), and suppose L is semi-homogeneous. Let $\rho(C)$ be the minimum rank of $C(x)$, and for each c, $\rho(C) \le c \le \min(r, n)$, let D_c be a minimal subset of Δ_c such that*

$$\Delta_c \subseteq \mathrm{Rad}(\Delta_{c+1}, D_c). \tag{10}$$

For any $\delta_{ab} \in D_c$, let S_{ab} be the corresponding special set. Then the family

$$\mathbf{S} = \{S_{ab} \mid \delta_{ab} \in D_c \text{ for some } c,\ \rho(C) \le c \le \min(r, n)\}$$

is an irredundant cover for L.

PROOF. Consider the finite family $\mathbf{T}$ of all special sets. Theorem 4.1 guarantees that this family covers L. By Lemma 3.1, we know that this family can be partitioned into a disjoint union of subfamilies $\mathbf{T}_c$ of special sets, where for each $\mathbf{T}_c$, all members have the same C-rank c, $0 \le c \le \min(r,n)$. Clearly, all special sets in $\mathbf{T}_c$ are empty if $c < \rho(C)$. Fix $c \ge \rho(C)$ and let $\Lambda_c(L)$ be the union of all members of $\mathbf{T}_c$. Thus $\Lambda(L) = \bigcup \Lambda_c(L)$. Let $\mathbf{S}_c = \{S_{ab} \mid \delta_{ab} \in D_c\}$. Applying the above corollary to the families $\mathbf{T}_c$ and $\mathbf{S}_c$ using (10), we have $\Lambda_c(L) = \bigcup S_{ab}$, where the union is taken over members of $\mathbf{S}_c$. Moreover, since L is semi-homogeneous, Proposition 4.4 implies $\pi(S_{ab}) = N(S_{ab})$. It follows that if S_{ab} were empty or redundant, by the above corollary, we could remove δ_{ab} from D_c and the resulting set would still satisfy (10), contradicting the minimality assumption. Thus each S_{ab} is a regime and $\mathbf{S}_c$ is irredundant. The proof is complete by Lemma 3.1. □

COROLLARY 6.5. *Let L be a PSLE as given by* (1) *and let $\rho(C)$, c, D_c (not necessarily minimal), and S_{ab}, be as in Theorem 6.4. Then the family*

$$\mathbf{S}' = \{S_{ab} \mid S_{ab} \neq \phi,\ \delta_{ab} \in D_c \text{ for some } c,\ \rho(C) \le c \le \min(r, n)\}$$

is a cover for L.

PROOF. Let $\mathbf{S}'_c$ be the subset of $\mathbf{S}_c$ consisting of all non-empty S_{ab}. The conclusion that $\Lambda_c(L) = \bigcup S_{ab}$ in the proof of Theorem 6.4 is still valid when the union is taken over $\mathbf{S}'_c$, without assumimg semi-homogeneity of L or minimality of D_c. □

The word "minimal" in the statement of Theorem 6.4 and Procedure 4 below may be interpreted either in the sense of set inclusion, or in the sense of cardinality. Unfortunately, in the present setting, there is no way to determine whether the cover is actually a minimum if "minimal" is interpreted in the second sense. We simply have not studied how solution functions may look if the regimes are not special. Nonetheless, Theorem 6.4 and the corollary provide a method for reducing the number of special regimes, which guarantees irredundancy in case L is semi-homogeneous. In the general case, while we may no longer get an irredundant cover from a minimal D_c, the number of special regimes is actually *less*, since we have to eliminate those that are empty. We describe this improved algorithm in terms of a procedure that computes D_c in Theorem 6.4. The implementation of this procedure will be discussed in the next section. Algorithm 2 below differs from Algorithm 1 by just two lines. When applied as in Algorithm 2, Procedure 4 amounts to exploring all relations between subdeterminants of $C(x)$ to avoid computing redundant solutions.

PROCEDURE 4. MinGenerator(g, h). Given families of polynomials $g = (g_1, \ldots, g_r)$, and $h = (h_1, \ldots, h_t)$, this procedure returns a minimal subfamily $H = (h_{i_1}, \ldots, h_{i_k})$ of h such that for every j, $1 \le j \le t$, h_j belongs to $\mathrm{Rad}(g, H)$.

ALGORITHM 2.

Input: A PSLE $L{:}C(x)z = A(x,w)$ as given by (1).
Output: A list of special regimes, together with their solution functions, satisfying the properties stated in Theorem 4.1. If L is semi-homogeneous, the list is irredundant.

STEP 1.
  For $c := \min(r,n) + 1$ to 0 by -1 do
    $D_c :=$ Determinants($C(x)$, c)
    if D_c consists of a single constant, go to STEP 2
    $D'_c :=$ MinGenerator(D_{c+1}, D_c)
    if HasSolution(D_c, 1) = $FALSE$, go to STEP 2

STEP 2.
  $\rho := \max(c, 0)$
  For $c := \min(r, n)$ to ρ by -1 do
    For $\delta_{ab}(x) \in D'_c$ do
      $h_{ab}(x, w) =$ Consistency(L, a, b)
      $\Phi_{ab} := D_{c+1} \cup \{h_{ab}(x,w)\}$
      if L is semi-homogeneous or HasSolution(Φ_{ab}, $\delta_{ab}(x)$) then
        $Z_{ab}(x,w) :=$ Solve(L, a, b)
        output Φ_{ab}, $\delta_{ab}(x)$, Z_{ab}.

PROOF OF CORRECTNESS. If L is semi-homogeneous, this follows from Theorem 6.4. In this case, there is no need to apply the test for non-emptyness using Procedure HasSolution in Step 2. Without the assumption of semi-homgeneity, we may have empty S_{ab} in $\mathbf{S}_c$, and the test eliminates these. □

7. Implementation Issues

In this section, we shall explore how the procedures specified in the previous sections may be implemented. We shall first consider how solution functions may be presented, and then discuss implementation strategies in turn for Procedures 3, 4, 1, and 2. As we saw in the last few sections, the solution to a PSLE is a list of solution functions (S, Z) where S is a quasi-algebraic set and Z is a matrix of rational functions defined on S. One of the main problems in implementation is to present each of these solution functions in the "simplest" form. The general problem for algebraic sets may be illustrated by an example.

EXAMPLE 7.1. Consider the case of two parameters x, y and let $h = \{x^2, xy, y^2\}$. The algebraic set $S = V(h)$ consists of the single point (0,0), and a simpler set of polynomials describing S is therefore $\{x, y\}$, which generates Rad(h). Let f be the polynomial function $x + y^2$ which is defined on S. One can obtain a simplified equivalent function $\hat{f}$ such that $f(\alpha, \beta) = \hat{f}(\alpha, \beta)$ for every $(\alpha, \beta) \in V(h)$ by reducing f modulo Rad(h). A less complete simplication may be obtained if we reduce f modulo Ideal(h) instead. In the present example, we may take $\hat{f} = 0$, but if we reduce only modulo Ideal(h) then we can only simplify f to x. □

The above example suggests that one way to carry out the simplification over algebraic sets is to be able to effectively solve two problems:

PROBLEM. 7.2. Compute a generating set for Rad(h), given h. □

PROBLEM. 7.3. Compute the reduction of f modulo Ideal(h), given f and h. □

As we shall see, both problems are solvable. The simplification problem for a quasi-algebraic set and rational functions defined on it, however, will be more involved, and is partially solved in Proposition 7.4 below.

We will digress briefly to review the concepts of term orderings and Gröbner bases. A *term ordering* is a total ordering $>$ of the set of monomials in $F[x]$ that is compatible with multiplication. Term orderings are essential in any computer representation of polynomial rings. A commonly used term ordering is the pure lexicographic ordering, implicitly given by the label order $x_m > x_{m-1} > \cdots > x_1$, or more precisely, $x_1^{e_1} x_2^{e_2} \ldots x_m^{e_m} > x_1^{d_1} x_2^{d_2} \ldots x_m^{d_m}$ if and only if $(e_m, \ldots, e_1)$ is lexicographically greater than $(d_m, \ldots, d_1)$. Another popular term ordering is the "total degree" term ordering on $F[x]$, which is obtained as a refinement of total degree by the pure lexicographic ordering. For a complete characterization of all term orderings, see Robbiano (1985), Dube *et al.* (1986), Weispfenning (1987), and Robbiano and Mora (1988). Kredel (1988) gave a survey of term orderings used in computer algebra systems. (However, the term orderings in SCRATCHPAD II reported by Kredel is incorrect, see Jenks' (1984) original report and Sit (1989)). Also, SCRATCHPAD II can now perform polynomial arithmetic relative to an arbitrary term ordering (in the domain Generalized Distributed Multivariate Polynomials).

Gröbner bases are defined relative to a fixed term ordering. They can be characterized in many ways (see Robbiano (1988), for example, where 12 equivalent conditions are given). Introduced by Buchberger in 1965, they have become fundamental tools used to attack many computational problems in algebra. We shall see that the two problems mentioned above can be solved using Gröbner bases. In addition, techniques for implementing all 4 procedures specified in the previous two sections are intimidly related to the simplification problem and will involve Gröbner bases computation too. For basic properties of Gröbner bases and their applications, the reader should consult Buchberger (1985, 1987) and other references cited there. More recently,

Mishra & Yap (1989) gave a self-contained exposition. In this paper, we follow Buchberger's original definitions. Given a finite set h of polynomials, and a fixed term ordering, a polynomial f is said to be in *normal form* (or *reduced*) if no monomials appearing in f is a multiple of any leading monomial of a polynomial in h. The set h is called a *Gröbner basis* if every polynomial in the polynomial ring reduces to a *unique* normal form modulo h. A Gröbner basis is said to be *reduced* if for all $f \in h$, f is in normal form modulo $h \backslash \{f\}$. Such a reduced Gröbner basis is unique (Theorem 6.3, Buchberger 1985). Given a finite set h, Buchberger gave an algorithm (Algorithm 6.3, *loc. cit.*) which computes a set h' of generators for Ideal(h) that is a reduced Gröbner basis. Given any Gröbner basis of an ideal (reduced or not), it is easy (Algorithm 6.1, *loc. cit.*) to compute the normal form of any polynomial, and a polynomial belongs to the ideal if and only if its normal form is zero. Most computer algebra systems provide routines for Gröbner bases computations.

Problem 7.3 can thus be effectively carried out by computing a Gröbner basis of Ideal(h) and then computing the normal form of f with respect to the basis. Problem 7.2 is a more expensive one to solve (see Gianni *et al.* (1988)). When Ideal(h) is zero-dimensional, it is relatively easy to compute its radical, using an algorithm of Seidenberg (1974), together with properties of Gröbner bases. The general case is computed by writing Ideal(h) as the intersection of two ideals, one with strictly smaller dimension, and the other in a polynomial ring of fewer indeterminates, and applying recursion to compute the radicals of these two ideals. Eventually, the recursion leads to the zero-dimension case. This algorithm has also been implemented on the SCRATCHPAD II system by Gianni (Gianni & Mora, 1989). There is currently much research on this problem, and alternate and less expensive algorithms may be at hand (see Alonso *et al.* (1990), Eisenbud *et al.* (1989), Giusti & Heintz (1990), Kobayashi *et al.* (1989), Krick & Logar (1990a, b), and Neff (1989); most of these references were supplied by one referee, who also pointed out that the algorithm of Eisenbud *et al.* was implemented in the Macaulay system). At present, in comparison to Gröbner bases routines, relatively few computer algebra systems have routines for radical computations. Thus, in this section, we shall take a dual approach. We shall continue to develop in theory how our algorithm is based on computations with radical ideals, while, in the discussions on implementation, we shall not insist on an available solution to Problem 7.2. Indeed, we shall show that without computing a Gröbner basis of a radical ideal, we can solve PSLE with possibly only a slight increase in the number of regimes and in the complexity of the representation of solution functions. We shall explore other heuristic ways using factorization to minimize these effects.

We now state the prerequisites on which an implementation may be based. We shall assume that routines in multivariate polynomial arithmetic, factorization, determinants, Gröbner basis, normal form, and optionally, radical computation are available. In these and other routines, wherever we use radical ideals, they may be replaced with ideals, with some sacrifice on simplicity. Since it is generally more expensive to compute the radical of an ideal, we can use the ideal version instead of the radical ideal version in performing simplifications at intermediate steps for efficiency.

ROUTINE 1R (resp. I). GröbnerBases(h). Given a finite set h of polynomials in a polynomial ring $F[x]$, this routine returns the *reduced* Gröbner basis of Rad(h) (resp. Ideal(h)); when Ideal(h) $= F[x]$, it returns the set $\{1\}$, and when Ideal(h) $= 0$ it returns the empty set. □

ROUTINE 2. NormalForm(G, f). Given a Gröbner basis G and a polynomial f, this routine returns the normal form $\hat{f}$ of f; in case the Gröbner basis consists of the empty set, it returns f itself. □

Straightly speaking, these routines (and others below) depend on fixing a polynomial ring and a term ordering. It is well-known that the complexity of the Gröbner bases algorithm is very sensitive to the ordering of the variables (Gebauer & Möller, 1988) when the purely lexicographical ordering is used, but nearly stable in the case of total degree ordering. An efficient method for transforming Gröbner bases with respect to different orderings in special cases has been recently discovered (Faugère *et al.*, 1988). However, since there is no known method to predict the optimal ordering in any given problem, we shall not concern ourselves with this aspect of the algorithm. Instead, we shall assume that these routines work for any user selected term ordering $>$, which we fix once and for all.

We now return to the simplification problem on algebraic and quasi-algebraic sets. Consider first the case of an algebraic set defined by a set h of polynomials. We shall regard the reduced Gröbner basis (with respect to $>$) of Rad(h) as a simplified set of defining polynomials of $V(h)$. For any polynomial function f defined on $V(h)$, we shall regard the normal form of f modulo Rad(h) as a simplified equivalent function. For any rational function p/q defined on $V(h)$ we shall regard $\hat{p}/\hat{q}$ as a simplified equivalent function, where $\hat{p}$ (respectively $\hat{q}$) is the normal form of p (respectively q) modulo Rad(h). Thus in this sense, Routines 1 and 2 may be viewed as simplifers on algebraic sets and functions defined on them.

Buchberger classified simplifiers as either canonical or non-canonical (see Buchberger, 1982). Since normal-form algorithms are canonical simplifiers (Buchberger, 1985, Method 6.1), it is natural to define "simplification" the way we did. It is possible, however, that the normal form $\hat{f}$ of a polynomial may have many more (but lower) monomial terms and with larger coefficients than the polynomial f itself. For other discussion on the simplification problem, see Lazard (1988), where he elaborated on how to compute the solutions of zero-dimensional algebraic systems in a specially simple form.

The concept of simplification for quasi-algebraic sets is more involved. We shall study only those quasi-algebraic sets that are of interest to us. Consider a quasi-algebraic set of the form $S = V^m(h) \cap \overline{V}^m(f)$, where h is as before, and f is a single polynomial in $F[x]$. We can construct an algebraic set $S^+ = V^{m+1}(h, tf-1) \subseteq U^{m+1}$, where t is a new indeterminate and h, and $tf-1$ are now polynomials in $F[x, t]$. Let $\pi: U^{m+1} \to U^m$ be the projection given by $\pi(a_1, \ldots, a_{m+1}) = (a_1, \ldots, a_m)$. Then π induces a bijection between S^+ and S. This bijection allows us to identify S with an algebraic set

S^+. The following proposition, which relates a Gröbner basis of Ideal$(h, tf-1)$ and Rad$(h, tf-1)$ with the quasi-algebraic set $S = V^m(h) \cap \overline{V}^m(f)$, provides a simplified description for S as well as for polynomial functions defined on S.

PROPOSITION 7.4. *With notations as above, let* $\overline{G}^+$ *be a Gröbner basis of* Rad$(h, tf-1)$ *(respectively,* Ideal$(h, tf-1)$*) relative to the order* $>_t$*, where* $t^i x^I >_t t^j x^J$ *if* $i > j$ *or if* $i = j$ *and* $x^I > x^J$*. Let* $\overline{G} = \overline{G}^+ \cap F[x]$ *and let* $\hat{f}$ *be a normal form of* f *modulo* $\overline{G}$*. Then*

(a) $\overline{G}$ *is a Gröbner basis for* Rad$(h, tf-1) \cap F[x]$ *(resp.* Ideal$(h, tf-1) \cap F[x]$*) with respect to* $>$.

(b) *If* $\overline{G}^+$ *is a reduced Gröbner basis, then so is* $\overline{G}$.

(c) *If* $p \in F[x]$*, then the normal form of* p *with respect to* $\overline{G}$ *is the same as the normal form* $\hat{p}$ *of* p *with respect to* $\overline{G}^+$.

(d) $\mathrm{Rad}_{F[x]}(\overline{G}) \supseteq \mathrm{Rad}_{F[x]}(h)$, *(resp.* $\mathrm{Ideal}_{F[x]}(\overline{G}) \supseteq \mathrm{Ideal}_{F[x]}(h)$*).*

(e) $S = V^m(\overline{G}) \cap \overline{V}^m(\hat{f})$.

PROOF. The proofs for both cases are similar; we give them together. Part (a) is just a special version of Proposition 3.1 (Gianni *et al.*, 1988), while part (b) follows easily from definition. Part (c) is clear (see Algorithm 6.1, Buchberger, 1985) since the normal form computation of p with respect to $\overline{G}^+$ cannot involve any member of $\overline{G}^+$ which is not in $F[x]$ because of the way we extend the term ordering. In particular, if $p \in \mathrm{Rad}(\overline{G}^+) \cap F[x]$ (resp. $p \in \mathrm{Ideal}(\overline{G}^+) \cap F[x]$), then $\hat{p} = 0$ and hence $p \in \mathrm{Rad}(\overline{G})$ (resp. Ideal$(\overline{G})$). Thus we have

$$\mathrm{Ideal}_{F[x]}(\overline{G}) = \mathrm{Ideal}_{F[x]}(\overline{G}^+ \cap F[x]) \supseteq \mathrm{Ideal}(\overline{G}^+) \cap F[x] \supseteq \mathrm{Ideal}_{F[x]}(h),$$

and $\mathrm{Rad}(\overline{G}) \supseteq \mathrm{Rad}(h)$, proving (d). In particular, $V^m(\overline{G}) \subseteq V^m(h)$. We observe that for any $a \in V^m(\overline{G})$, $f(a) \neq 0$ if and only if $\hat{f}(a) \neq 0$. Now let $a \in S$, $a_{m+1} = 1/f(a)$, and $a^+ = (a, a_{m+1})$. Then $a^+ \in S^+$, and hence $g^+(a^+) = 0$ for all $g^+ \in \overline{G}^+$. For any $g \in \overline{G}$, $g(a^+) = g(a) = 0$. Thus $a \in V^m(\overline{G})$ and by our observation, $\hat{f}(a) \neq 0$ and so $S \subseteq V^m(\overline{G}) \cap \overline{V}^m(\hat{f})$. Since $V^m(\overline{G}) \subseteq V^m(h)$, the converse also follows from the observation, proving (e). □

Part (c) of the proposition suggests a way to simplify a polynomial function defined on a quasi-algebraic set, namely, compute its normal form with respect to $\overline{G}$. Similarly, Part (e) yields a representation for S which may be viewed as "simplified." The specific term ordering $>_t$ described in the proposition must be adhered to for parts (c) and (e) to be valid. When the given term ordering $>$ on $F[x]$ is the pure lexicographic ordering, the extended term ordering $>_t$ is just the pure lexicographic ordering on $F[x, t]$ (implicitly: $t >_t x_m >_t \ldots >_t x_1$). This property makes implementation using the pure lexicographic ordering much easier. In contrast, the extension $>_t$ of the total degree term ordering on $F[x]$ is neither the total degree term ordering on $F[t, x]$ nor the total degree term ordering on $F[x, t]$. For example, $tx_1^2 >_t x_2^3$ and $t >_t x_1^2$, but in the case of total degree term ordering on $F[t, x]$, x_2^3 is greater than tx_1^2 while in the case of total degree term ordering on $F[x, t]$, x_1^2 is greater than t. A similar phenomenon occurs

when the original term ordering is a refinement of total degree by the reverse lexicographic ordering (that is, $x_1^{e_1}x_2^{e_2}\dots x_m^{e_m} > x_1^{d_1}x_2^{d_2}\dots x_m^{d_m}$ if and only if $(\sum_{i=1}^{m} e_i, -e_1, \dots, -e_m)$ is lexicographically greater than $(\sum_{i=1}^{m} d_i, -d_1, \dots, -d_m)$).

There are other limitations. We illustrate below with examples to show that Part (e) does not always give the "simplest" representation of S.

EXAMPLE 7.5. Let x, y be two indeterminates, $h = \{xy^2\}$, and $f = x^2$. Then S is the x-axis minus the origin. Using a term ordering with $t > x > y$, a Gröbner basis for Ideal$(h, tf-1)$ is $\overline{G}^+ = \{y^2, tx^2-1\}$. Thus $\overline{G} = \overline{G}^+ \cap F[x,y] = \{y^2\}$, and Ideal$(\overline{G})$ properly contains Ideal(h). We see, in general, Ideal$(\overline{G}) \neq$ Ideal(h). This same example also shows that Rad$(\overline{G}) \neq$ Rad(h). In addition, note that the Gröbner basis $\overline{G}$ obtained through Rad$(h, tf-1)$ is $\{y\}$, which is simpler than the one above obtained via Ideal$(h, tf-1)$. On the other hand, $\hat{f} = x^2$ in both the Ideal and Rad cases, but the simplest description of S is $V(y) \cap \overline{V}(x)$. Thus Part (e) is only a partial answer to this simplification problem. □

EXAMPLE 7.6. Consider the case of three *real* parameters x, y, and z. Let $h = \{xz+y, x-yz\}$ and let $f = z$. It is not difficult to show that the quasi-algebraic set S so defined is the z-axis with the origin deleted and a simple description is given by $V(x,y) \cap \overline{V}(z)$. With a term ordering $t > x > y > z$, the Gröbner basis $\overline{G}^+ = \{ty+yz, tz-1, x-yz, yz^2+y\}$ and $\overline{G} = \{x-yz, yz^2+y\}$. Thus Part (e) yields a description of S as $V(x-yz, yz^2+y) \cap \overline{V}(z)$. The difficulty here is of course in recognizing that $z^2+1 \neq 0$ for the real parameter z. This example suggests that perhaps one may solve this simplification problem by viewing it as a special class of elementary algebra and geometry quantifier elimination problem. However, as Arnon and Mignotte (1988) pointed out, the quantifier elimination problem for (the more general) semi-algebraic sets has the same difficulty in defining what is "simplest." □

With these limitations in mind, we may now specify a simplifier for quasi-algebraic sets based on Proposition 7.4. Routine 3 is a generalization of Routine 1, and in the spirit of SCRATCHPAD II, we shall refer to Routine 3 with the same name – the two can be distinguished by their arguments. Again, Routine 3 may be specified relative to either ideals or radical ideals. It necessitates the introduction of an additional indeterminate t, and can be implemented using Routines 1 and 2. We note that there is no need for an extra routine to simplify polynomial functions on quasi-algebraic sets since by Part (c) of Proposition 7.4, Routine 2 may be used, provided $\overline{G}$ is a Gröbner basis as first obtained using Routine 3.

ROUTINE 3R (resp. I). GröbnerBases(h, f). Given a finite set h of polynomials, and a polynomial f in $F[x]$, this routine returns a reduced Gröbner basis $\overline{G}$ and the normal form $\hat{f}$ of f with respect to $\overline{G}$ as described by Proposition 7.4, radical ideal (resp. ideal) version. In other words, given a quasi-algebraic set $S = V(h) \cap \overline{V}(f)$, it returns a simplified description $S = V(\overline{G}) \cap \overline{V}(\hat{f})$. When $f = 1$ this reduces to Routine 1R (resp. I). □

The ideal version of Routine 3 is particularly useful as the following well-known result shows.

PROPOSITION 7.7. *With notations as in Routine 3I, the following are equivalent.*

(a) $S = \phi$,
(b) $f \in \text{Rad}(h)$,
(c) $1 \in \overline{G}$,
(d) $\hat{f} = 0$.

PROOF. The proposition is trivial if $f = 0$. Suppose $f \neq 0$. We have (a) holds if and only if f vanishes identically on $V(h)$, which is equivalent to (b) by Hilbert's Nullstellungsatz. Let t be a new indeterminate. We claim that $f \in \text{Rad}_{F[x]}(h) \Leftrightarrow f \in \text{Rad}_{F[x,t]}(h, tf - 1)$: for if for some k, f^k can be written as a linear combination of h_i and $tf - 1$ with coefficients in $F[x, t]$, then by substituting $1/f$ for t and clearing denominators, we obtain for some k', $f^{k'}$ as a linear combination of h_i with coefficient in $F[x]$. Thus, using the notations of Proposition 7.4 (ideal version), we have

$$\text{(b)} \Leftrightarrow 1 \in \text{Ideal}(h, tf - 1) \Leftrightarrow 1 \in \overline{G}^+ \Leftrightarrow \text{(c)}.$$

Since $\overline{G}$ is a Gröbner basis of $\text{Ideal}(h, tf - 1) \cap F[x]$, it is clear that (c) and (d) are equivalent. □

By this proposition, Routine 3I can be used to implement Procedure 3 since $\text{HasSolution}(h, f) = FALSE \Leftrightarrow S = \phi$. It can also be used to decide the membership problem for a radical ideal, *without first computing a Gröbner basis for the radical ideal.* Routine 2 may be used to solve the membership problem for ideals.

A certain amount of simplification can be obtained using factorization and some heuristics, without adding an extra indeterminate and hence overheads because of dual representations of the original polynomials. Factorization of multivariate polynomials of course is non-trivial in general, but most computer algebra systems have such routines. Routine 3F below may be viewed as another partial simplifier, and if desired, may be used repeatedly. The following notations will be used. Given a polynomial p in $F[x]$, we let $\xi(p)$ denote the set of all distinct factors of p irreducible over F. Given a set ξ of distinct irreducible polynomials in $F[x]$ we let ξ^* denote their product. As usual, $\xi^* = 1$ if ξ is empty. If ξ' is another set, we let $\xi \backslash \xi'$ denote the set of irreducible polynomials in ξ but not in ξ'. In the description, comment are delimited by "/*" and "*/."

ROUTINE 3F

Input: A quasi-algebraic set $S = V(h) \cap \overline{V}(f)$ where $h = (h_1, \ldots, h_s) \in F[x]^s$ and $f \in F[x]$.
Output: A Gröbner basis $\overline{G}$ and a polynomial $\hat{f}$ reduced with respect to $\overline{G}$ such that $S = V(\overline{G}) \cap \overline{V}(\hat{f})$.

Algorithm:

$G_0 :=$ GröbnerBases(h) /* *Routine 11* */
$f_0 :=$ NormalForm(G_0, f) /* *Routine 2* */
If $f_0 = 0$ then output $\overline{G} = \{1\}, \hat{f} = 0$ and stop.
$\Xi_1 := \{\xi(g) \mid g \in G_0$ and $\xi(g) \supset \xi(g')$ for any $g' \in G_0\}$
$\Xi_2 := \{\xi(g) \backslash \xi(f_0) \mid \xi(g) \in \Xi_1\}$
$G_2 := \{\xi^* \mid \xi \in \Xi_2\}$
$\overline{G} :=$ GröbnerBases(G_2) /* *Routine 11* */
$\hat{f} :=$ NormalForm($\overline{G}, \xi(f_0)^*$) /* *Routine 2* */
Output $\overline{G}, \hat{f}$.

PROOF OF CORRECTNESS. Since Ideal(G_0) = Ideal(h) it is clear that $S = V(G_0) \cap \overline{V}(f_0)$. If $f_0 = 0$ then $S = \phi$ and we stop with the appropriate output. Assume now $f_0 \neq 0$. We observe that for any non-zero polynomial p, Rad(p) = Rad($\xi(p)^*$), $V(p) = V(\xi(p)^*)$, and $\overline{V}(p) = \overline{V}(\xi(p)^*)$. This observation extends to a family of polynomials as well. Let $G = \{\xi(g)^* \mid g \in G_0\}$. Then we have Rad($G_0$) = Rad($G$). If for some g, $g' \in G_0$ we have $\xi(g) \supset \xi(g')$ then $\xi(g')^*$ properly divides $\xi(g)^*$. Thus if $G_1 = \{\xi^* \mid \xi \in \Xi_1\}$, then Rad($G_1$) = Rad($h$) and hence $V(G_1) = V(h)$. Let $g \in G_1$ and let p be an irreducible common factor of f_0 and g. Clearly, if $\alpha \in S$ then $p(\alpha) \neq 0$ and hence $(g/p)(\alpha) = 0$. This shows that $S = V(G_2) \cap \overline{V}(f_0)$. Since Ideal($G_2$) = Ideal($\overline{G}$) and $\overline{V}(f_0) = \overline{V}(\xi(f_0)^*)$ it follows that $S = V(\overline{G}) \cap \overline{V}(\hat{f})$. □

EXAMPLE 7.8. Let $S = V(xy^2) \cap \overline{V}(x^2)$ as in Example 7.5. Routine 3F yields $S = V(y) \cap \overline{V}(x)$. In this example, Routine 3F gives better result than Routine 3R and Routine 3I. □

EXAMPLE 7.9. Let x, y be two indeterminates ordered lexicographically with $x > y$. Let $h = (x^2 + y, y^2)$ and $f = x$. Then $S = V(h) \cap \overline{V}(f) = \phi$. Following Routine 3F we get $G_0 = h, f_0 = f$, $G_1 = \{x^2 + y, y\} = G_2$, and finally $\overline{G} = \{x^2, y\}$, and $\hat{f} = x$. Applying Routine 3F once more yields $\overline{G} = \{1\}$. This example shows Routine 3F is only a partial simplifier. It also shows that the analogue of Proposition 7.7 does not hold (except the equivalence of (a) and (b)) and hence cannot be used to implement Procedure 3. □

We now turn our attention to the implementation of Procedure 4: MinGenerator. Keeping in mind that MinGenerator is applied in Algorithm 2 with inputs Δ_{c+1}, Δ_c for $\rho(C) \leq c \leq \min(r, n)$, and we are interested in simplifying functions defined on quasi-algebraic subsets of $V(\Delta_c)$, we would like, in addition to a minimal subset D_c satisfying equation (10), to have a Gröbner basis for Rad(Δ_c) or Ideal(Δ_c). In terms of the inputs $g = (g_1, \ldots, g_s)$ and $h = (h_1, \ldots, h_q)$ of MinGenerator, we require also a Gröbner basis for Rad(g, h) or Ideal(g, h).

We shall study several choices, none of them completely satisfactory. Because of the complexity of Buchberger's algorithm, it is difficult to devise a computation model to compare the efficiencies of these alternatives. In the following discussions, we shall frequently compute a Gröbner basis for the *ideal* of an input family of polynomials that consists of a Gröbner basis and one extra polynomial. For convenience, we shall refer

to such a computation as an FGB (for fast Gröbner basis) computation. If a Gröbner basis is constructed from k input polynomials recursively, we may roughly consider that as equivalent to k FGB computations. For example, computing a Gröbner basis for Ideal(g) involves s FGB computations, which is fixed and independent of q. It should be noted that the time for an FGB computation is not constant and still depends on the actual input. Nonetheless, comparing the number of FGB computations (as a function of q) in the alternatives at least provides an intuitive idea of their relative efficiencies. We shall not provide further analysis beyond this, but shall point out the advantages and disadvantages of each.

Our first method is the brute force approach, that is, for every subfamily (or subset) H of h we use Routine 3I to test if each $h_i \notin H$, $1 \le i \le q$, belongs to Rad(g, H); among all H for which this is true for all i we select one with a minimum number of polynomials. This of course guarantees minimality, but is rather inefficient, since there is clearly a lot of redundant computation. Some redundancy may be avoided by using a clever enumeration algorithm (for example, the LEXSUB algorithm in Nijenhuis & Wilf (1978)) for all subsets of h so that most calls to Routine 3I are FGB computations. Since there are 2^{q-1} subsets not containing a particular h_i, the brute force method may require $q2^{q-1}$ FGB computations (ignoring any fixed costs) in the worst case. However, even though we obtain a minimum subset H such that Rad(g, H) = Rad(g, h), in general we have not computed a Gröbner basis for this radical ideal; indeed, not even one for Ideal(g, h) (Example: g is the empty family, and $h = \{x^2, xy, y^2\}$).

Next, we shall present 3 approximate methods, each a variant on the greedy algorithm. Routine 4M involves computing Gröbner bases of ideals and membership testing for radical ideals. Routine 4I involves only computing Gröbner bases of ideals, while Routine 4R involves computing Gröbner bases of radical ideals. The inputs are the same, and the algorithms are sensitive to the order in which the polynomials h_i is given. Let $I_0 = \text{Ideal}(g)$ and for $1 \le i \le q$ let $I_i = \text{Ideal}(g, h_1, \ldots, h_i)$. Let R_0 and R_i be the corresponding radical ideals. They all compute a "near minimal" subset H_q of h such that $\text{Rad}(g, H_q) = R_q$.

ROUTINE 4M

Additional Output: A Gröbner basis G_q for I_q.
Algorithm:

```
G_0 := GröbnerBases(g)            /* Routine 1I */
H_0 := φ
For i = 1 to q do
   (G_i, ĥ_i) := GröbnerBases(G_{i-1}, h_i)      /* Routine 3I */
   if ĥ_i = 0, then
      H_i := H_{i-1}
   else
      H_i := H_{i-1} ∪ {h_i}
   G_i := GröbnerBases(G_{i-1} ∪ {h_i})          /* Routine 1I */
Output G_q, H_q.
```

PROOF OF CORRECTNESS. It is clear that G_i is a Gröbner basis of I_i. We shall prove by induction that $\text{Rad}(g, H_i) = R_i$. For $i = 0$ there is nothing to prove. For $i \geq 1$, consider first the case $\hat{h}_i = 0$. By Proposition 7.7, and our induction hypothesis, $h_i \in \text{Rad}(G_{i-1}) = R_{i-1} = \text{Rad}(g, H_{i-1})$. Hence $R_i = R_{i-1} = \text{Rad}(g, H_{i-1}) = \text{Rad}(g, H_i)$. Next consider the case $\hat{h}_i \neq 0$. Then $\text{Rad}(g, H_i) = \text{Rad}(g, H_{i-1} \cup \{h_i\}) = \text{Rad}(R_{i-1} + \text{Rad}(h_i)) = \text{Rad}(I_{i-1} + \text{Ideal}(h_i)) = R_i$. □

Routine 4M requires $2q$ FGB computations, thus is much more efficient than the brute force method. It also yields a Gröbner basis for I_q. It should be noted that we have to compute a new Gröbner basis for I_i even in case $\hat{h}_i = 0$ since h_i need not be in I_{i-1}. The Gröbner bases $\overline{G}_i$ have no subsequent use. This method does not guarantee a minimum subfamily. Note also that $\text{Ideal}(g, H_q) \subseteq I_q$, but the inclusion may be strict (see Example 7.10 below).

EXAMPLE 7.10. Let g be the empty family and let $h_1 = xy^2$, $h_2 = x^2$ and $h_3 = x$. Following Routine 4M, we obtain $G_1 = H_1 = \{h_1\}$, $G_2 = H_2 = \{h_1, h_2\}$, and $\overline{G}_3 = \{1\}$. Thus $G_3 = \{h_3\}$ and $H_3 = H_2$. Note that $\text{Ideal}(H_3) \neq I_3$ and so we must compute a new Gröbner basis by including h_3. Note also that H_3 is not minimum, and that if we had sorted the input polynomials by total degree, we would have gotten the minimum. □

ROUTINE 4R

Additional Output: A Gröbner bases G_q of R_q.
Algorithm:

```
G_0 := GröbnerBases(g)              /* Routine 1R */
H_0 := φ
For i = 1 to q do
    ĥ_i := NormalForm(G_{i-1}, h_i)     /* Routine 2 */
    if ĥ_i = 0, then
        G_i := G_{i-1}
    H_i := H_{i-1}
    else
        G_i := Gr.Bases(G_{i-1} ∪ {h_i})    /* Routine 1R */
        H_i := H_{i-1} ∪ {h_i}
Output G_q, H_q.
```

Routine 4R is obviously correct, but it does not guarantee a minimum H_q and therefore has little advantage over Routine 4M. It requires q computations of Gröbner bases of radical ideals (not counting the one for G_0), and even though for each computation, the input consists of a Gröbner basis and one extra polynomial, it is no simple analogue of an FGB computation.

ROUTINE 4I

Output: A subset H_q of h such that $\text{Ideal}(g, H_q) = I_q$ (*a fortiori*, $\text{Rad}(g, H_q) = R_q$) and a Gröbner basis G_q for I_q. When g_j and h_i are all homogeneous, and the polynomials in h are arranged in non-decreasing order by total degree, H_q is actually a minimum subset.

Algorithm:

```
G_0 := GröbnerBases(g)                    /* Routine 11 */
H_0 := φ
For i = 1 to q do
   ĥ_i := NormalForm(G_{i-1}, h_i)        /* Routine 2 */
   if ĥ_i = 0, then
      G_i := G_{i-1}
      H_i := H_{i-1}
   else
      G_i := GröbnerBases(G_{i-1} ∪ {ĥ_i})   /* Routine 11 */
      H_i := H_{i-1} ∪ {h_i}
Output G_q, H_q.
```

PROOF OF CORRECTNESS. It is clear by induction that G_q is a Gröbner basis of I_q and $I_q = \text{Ideal}(g, H_q)$. Suppose now that for $1 \le i \le q$, h_i is homogeneous of degree d_i and these are arranged in non-decreasing order by total degree. To prove that H_q is minimal, we proceed by induction on q. The case $q = 1$ is trivially true. Assume by induction that H_{q-1} is minimal for I_{q-1}. Let K be a minimal subset of $\{h_1, \ldots, h_q\}$ such that $\text{Ideal}(g, K) = I_q$. Suppose first that $h_q \notin K$. Then $K \subseteq \{h_1, \ldots, h_{q-1}\}$ and $I_{q-1} = I_q$. In particular, $\hat{h}_q = 0$, and $h_q \notin H_q$. Thus $H_q = H_{q-1}$ and our induction hypothesis shows that $|H_{q-1}| = |K|$, where $|A|$ denotes the cardinality of a set A.

Next, we suppose that $h_q \in K$. Let $K' = K \backslash \{h_q\}$. Clearly, $\text{Ideal}(g, K') \subseteq I_{q-1}$. For any i, $1 \le i \le q-1$, we have $h_i \in I_q = \text{Ideal}(g, K)$. Thus we can express h_i as a sum of an element in I_0 and a linear combination of elements of K with coefficients which are polynomials. By writing these coefficients as sums of their homogeneous parts, and eliminating all products (formed from such parts and elements of K) of degree not equal to d_i, we see that h_i can be written as a sum of the form

$$h_i = \theta + \sum_{k \in K} p_{i,k} k \qquad (1 \le i \le q-1), \tag{11}$$

where $\theta \in I_0$ is homogeneous of degree d_i or zero, and for all $k \in K$, $p_{i,k}$ is either zero or homogeneous of degree $d_i - \text{degree}(k) \ge 0$. In this relation, the coefficient p_{i,h_q} must be either zero or is non-zero and of degree 0 (in which case, $d_i = d_q$). We observe that if for all i $(1 \le i \le q-1)$ we have $p_{i,h_q} = 0$, then $\text{Ideal}(g, K') = I_{q-1}$. On the other hand, if for at least one i, $p_{i,h_q} \ne 0$, then $h_q \in I_{q-1}$ and $h_q \notin H_q$.

Now either $h_q \in H_q$ or $h_q \notin H_q$. In the first case, our observation shows that $I_{q-1} = \text{Ideal}(g, K')$; by induction, $|K'| \geq |H_{q-1}|$ and hence $|K| = |H_q|$. In the second case, $I_q = I_{q-1}$ and the minimality of K shows that we cannot have $\text{Ideal}(g, K') = I_{q-1}$. Thus there is an h_l for which $p_{l,h_q} \neq 0$. Let $K'' = K' \cup \{h_l\}$. By equation (11), $\text{Ideal}(g, K'') = I_q = I_{q-1}$. Since $K'' \subseteq \{h_1, \ldots, h_{q-1}\}$, it follows that $|H_q| = |H_{q-1}| \leq |K''| = |K|$. This completes the proof. □

Routine 4I requires at most q FGB computations (not counting the initial computation of G_0). It has the advantage that it yields a Gröbner basis G_q for I_q, a generating subset H_q such that $\text{Rad}(g, H_q) = R_q$, and guarantees minimality (with respect to $\text{Ideal}(g, H_q) = I_q$) for homogeneous inputs. The algorithm (when g is the empty family) was given in a tutorial by Stillman (1986) without proof. This proof has been included because the author was unable to locate any in the literature. We note that the proof on minimality made use of no properties of Gröbner bases. Indeed, Gröbner bases were used only to provide a unique normal form for h_l, or equivalently, to provide an algorithm to test membership of h_l in the ideal I_{l-1}. Contrary to the ideal version, the radical ideal version (Routine 4R) will not, in general, compute a minimum subset H_q even when the inputs are homogeneous. Routine 4I is more efficient than Routine 4M, and does not require the introduction of a new indeterminate t. On the other hand, Routine 4M may yield a smaller generating set for R_q.

The following example illustrates these routines and shows that they do not always provide a minimum generating set in the non-homogeneous case.

EXAMPLE 7.11. Let g be the empty family and let $h_1 = xy$, $h_2 = x(1+y^2)$, and $h_3 = x^2y^4 - (1+y^2)$. The polynomials are already in non-decreasing order (by total degree, or any term ordering). Following the routines (any version), we have $\hat{h}_2 = x$ and $\hat{h}_3 = -(1+y^2)$. Thus the routines all yield $H_3 = \{h_1, h_2, h_3\}$ whereas $H = \{h_1, h_3\}$ is a minimum generating subset for I_3 as well as for R_3. Homogenizing this example, with $f_1 = xy$, $f_2 = x(z^2+y^2)$, and $f_3 = x^2y^4 - (z^2+y^2)z^4$, the routines still yield $K = \{f_1, f_2, f_3\}$ as the generating set. Now for Routine 4I, K is the minimum generating set of the ideal I_3, as proved. On the other hand, since

$$\begin{aligned} f_2^3 &= x^3(z^2+y^2)^2(z^2+y^2) \\ &\equiv x^2z^4(z^2+y^2) \quad \text{modulo } \text{Ideal}(f_1, f_3) \\ &\equiv 0 \quad \text{modulo } \text{Ideal}(f_1, f_3) \end{aligned}$$

and so $f_2 \in \text{Rad}(f_1, f_3)$, all versions fail to produce the minimum generating set $\{f_1, f_3\}$ of the radical ideal R_3. Finally, dehomogenizing K will not give the required minimum H. □

Thus using Routine 4 (any version) instead of MinGenerator in the algorithm for PSLE may result in more redundancy. There are several alternatives to remedy this situation. First, we can apply Routine 4 and then continue with the brute force approach. In case there are still redundant regimes (which is possible if L is not semi-

homogeneous), one may (relatively easily) further refine the output list by applying Proposition 6.2 to pairs of regimes S_{ab}, $S_{a'b'}$ with the same C-rank. For suppose, after applying Routine 4, we find that

$$S_{ab} = V(G_{ab}) \cap \overline{V}(\hat{\delta}_{ab}), \qquad S_{a'b'} = V(G_{a'b'}) \cap \overline{V}(\hat{\delta}_{a'b'}), \qquad G_{ab} = G_{a'b'}.$$

We can test redundancy by the following criterion:

$$S_{ab} \subseteq S_{a'b'} \Leftrightarrow \delta'_{ab} \in \mathrm{Rad}(G_{a'b'}, \hat{\delta}_{a'b'}) \Leftrightarrow 1 \in \mathrm{GröbnerBasis}(G_{a'b'}, \hat{\delta}_{a'b'}, t\hat{\delta}_{ab} - 1),$$

where t is a new indeterminate. Example 8.2 illustrates this possibility.

Despite its theoretical weakness (in the non-homogeneous cases and radical ideal version), when Routine 4I is used in place of MinGenerator, with the family h sorted in ascending order of total degree, it almost always yields the minimum generating sets in our test cases. Using Routine 4M with the same input order improves results but at twice the cost.

For Procedure Determinants, we find it harder to implement efficiently. Ideally, we would like to compute determinants of all square submatrices of $C(x)$ only once, and compute higher order determinants from lower order ones. Algorithms for determinants, like the Gauss-Bareiss reduction, compute *some* of subdeterminants along the way. However, in order to guarantee that the list of regimes covers L it may be necessary to compute *all* determinants of a certain size, as the example below shows.

EXAMPLE 7.12. Referring to Example 2.1, we note that the two determinants computed by Gauss-Bareiss reduction are δ_1 and δ_2. The ideal they generate is Ideal(a, b), which is a radical ideal, and does not contain $\delta_3 = 1$. Thus there seems to be no way to reduce the number of determinant computations. Unless by luck we happen to compute δ_3 first, all three 2×2 determinants have to be computed. □

For many PSLE, the lower order determinants are not needed (for example, when the order is less than the minimum rank $\rho(C)$). To determine $\rho(C)$, Proposition 4.2 suggests computing in the descending order by rank. An added advantage of this iteration order is that when Routine 4 is applied within the loop of Step 1 of Algorithm 2, the initial computation G_0 is simply the output G_q of the previous iteration. (That is why we did not count these fixed costs earlier!) Once this order of iteration is chosen, we find that the simplest way (though not in any sense efficient) to implement Procedure Determinants is by brute force, namely, by iterating through a double loop over the lists of all c-subsets of $\{1, \ldots, r\}$ and of $\{1, \ldots, n\}$. Such lists may in turn be generated by a simple algorithm (see Nijenhuis and Wilf, 1978, for example) and most computer algebra systems already have routines for determinants based on some elimination scheme. This need not be as bad as it sounds. The alternative would be to adapt those routines to generate subdeterminants for non-square matrices. Unless there are some theoretical guarantee that *all* subdeterminants have been computed, one would have to, at the very least, keep track of what subdeterminants are computed. It is not

clear how to interface such bookkeeping with any existing implementation of computing determinants (over multivariate polynomial rings) and there would be the problem of generating the rest of the subdeterminants. The theoretical weakness of the greedy algorithm and this dilemma suggest that to find a better implementation strategy may require us to treat the entire Step 1 of Algorithm 2 as a single problem. We summarize this below and leave this for future research.

OPEN PROBLEM. Let $C(x)$ be an $r \times n$ matrix with entries in a polynomial ring $R[x]$, where x is an m-dimensional vector of indeterminates. Let I^c be the ideal (resp. let R^c be the radical ideal) generated by all determinants of $c \times c$ submatrices of $C(x)$. How can we efficiently compute $\rho(C)$ and for each c, $\rho(C) \leq c \leq \min(r,n)$, a minimum list D_c of determinants of $c \times c$ submatrices which generates I^c modulo I^{c+1} (resp. R^c modulo R^{c+1})? □

The remaining procedures to be addressed are Consistency and Solve. Here, again, most computer algebra systems have routines to solve linear systems with coefficients in polynomial rings. It is convenient to simply use these to find h_{ab} and Z_{ab} for every $\delta_{ab} \in D_c$. On the other hand, we would be duplicating some of the computations done during Procedure Determinants. For *small* r and n, it may be more efficient to keep these determinants in memory, and implement Procedures Consistency and Solve by applying the fact that $K_{ab}(x)$ is the adjoint of $C^{ab}(x)$ divided by $\delta_{ab}(x)$ and the adjoint can be obtained by looking up $(c-1) \times (c-1)$ determinants (*note*: Procedure Determinants will have to be modified slightly). Then h_{ab} and Z_{ab} may be computed, using mostly polynomial arithmetic, by equations (6) and (5).

The output of Algorithm 2 for PSLE consists of the polynomials defining the regimes S (equation (7)), and the solution functions Z. The representation of the regimes may be simplified by Routine 3I, which can make use of a Gröbner basis of I^{c+1} returned after applying Routine 4I (or 4M). For example, we can compute a simplified representation of $N(S)$ (equation (9)) using only one FGB computation. The consistency functions can then be simplified and added to obtain a representation of S. Finally, the entries in the solution functions can be simplified, again by Routine 3I. We like to point out that while the frequency and choice of routines in simplification are more an art than a science, the ideal versions work well enough for intermediate results.

8. SCRATCHPAD Examples

In our first implementation, on IBM's SCRATCHPAD II computer algebra system, we used a combination of Routines 3F, 3I for simplification and 4I for minimization. The result is a close version of Algorithm 2. The package is named PLEQN (for

Parametric Linear Equations) and uses the Distributed Multivariate Polynomial (DMP) representation for polynomials. (An updated version is now available for use with any polynomial ring $R[x]$ over a GCD domain R and is independent of the internal representation of $R[x]$.) In this first implementation, we have not used the routines for radical ideals, thus some of the results may not be as simple as they can be. The main exported function from PLEQN is called *psolve* (for parametric solve) and this function may be referenced, in the simplest case, by a user-supplied coefficient matrix $C(x)$ and a right-hand side vector $A(x, w)$. (In the homogeneous case, $A(x, w)$ may be omitted.) The output from *psolve* is a list of solution functions (S,Z). Each regime S is described by a list Φ of polynomials that vanish on the set and a list Ψ of polynomials that are non-zero at all points on the set. The list Φ is further separated into Φ_1 and Φ_2 where Φ_1 involves only x, and Φ_2 involves x and w. The list Ψ is a list of square-free factors of $\delta(x)$ (simplified). The particular solution comes from simplifying Z_0 and the basis comes from simplifying the remaining columns of the special solution function Z.

The following is a typical SCRATCHPAD II session running on an IBM 3090 using PLEQN, where four PSLEs were solved. In SCRATCHPAD II, L, E, RN, DMP, M, V, and SE are respectively abbreviations for the domains List, Expression, Rational Numbers, Distributed Multivariate Polynomial, Matrix, Vector, and Sorted Expressions. A semicolon at the end of an input statement suppresses output (usually just echo) from the system. A colon or double colon signifies domain association and may be read as "belongs to". Outputs from SCRATCHPAD II are indented.

In these examples, $F[x]$ is the polynomial ring *polring* $= \mathbb{Q}[d, c, b, a]$. The term ordering is pure lexicographical order with $a > b > c > d$ and the representation is DMP. The list Φ_1 is labelled as *eqzro*, Φ_2 as *wcond*, and Ψ as *neqzro*. The particular solution *partsol* and basis *basis* are given in a record called *bsoln*.

EXAMPLE 8.1. This is the generic 2×2 system considered in Example 2.2 (compare output with Figure 1).

```
r := 2; n := 2;
parm : L E := [a, b, c, d];
polring := DMP(parm, RN);
Coeff : M polring := zero(r, n);
Coeff(0) := [a, b] :: V polring;
Coeff(1) := [c, d] :: V polring;
Coeff
```

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

```
Avector : L SE := [u, v];
psolve(Coeff, Avector)

 [ [eqzro = [a, b, c, d], neqzro = [ ], wcond = [v, u],

   bsoln = [partsol = [0, 0], basis = {[0, 1],[1, 0]}] ]
```

$[eqzro = [ad - bc], neqzro = [d], wcond = [-bv + du],$

$bsoln = [partsol = [0, \frac{v}{d}], basis = \{[1, -\frac{c}{d}]\}]]$

$[eqzro = [ad - bc], neqzro = [c], wcond = [-av + cu],$

$bsoln = [partsol = [\frac{v}{c}, 0], basis = \{[-\frac{d}{c}, 1]\}]]$

$[eqzro = [ad - bc], neqzro = [b], wcond = [bv - du],$

$bsoln = [partsol = [0, \frac{u}{b}], basis = \{[1, -\frac{a}{b}]\}]]$

$[eqzro = [ad - bc], neqzro = [a], wcond = [av - cu],$

$bsoln = [partsol = [\frac{u}{a}, 0], basis = \{[-\frac{b}{a}, 1]\}]]$

$[eqzro = [\,], neqzro = [ad - bc], wcond = [\,],$

$bsoln = [partsol = [\frac{-bv + du}{ad - bc}, \frac{av - cu}{ad - bc}], basis = \{\}]]]$ □

EXAMPLE 8.2. We find the set of equilibrium points of the following Lotka Volterra system studied by Gardini *et al.* (1987):

$$\dot{z}_1 = z_1(1 - z_1 - az_2 - bz_3),$$
$$\dot{z}_2 = z_2(1 - bz_1 - z_2 - az_3),$$
$$\dot{z}_3 = z_3(1 - az_1 - bz_2 - z_3),$$

where we have replaced the original parameters α, β by a, b respectively. The non-trivial equilibrium points are found by solving the parametric linear system obtained by setting the right hand sides equal to zero (Examples 3.3 – 3.5).

```
r := 3; n := 3;
Coeff: M polring := zero(r, n);
Coeff(0) := [1, a, b] :: V polring;
Coeff(1) := [b, 1, a] :: V polring;
Coeff(2) := [a, b, 1] :: V polring;
Avector := [1, 1, 1] :: V polring;
psolve (Coeff, Avector)
```

$[[eqzro = [a - 1, b - 1], neqzro = [\,], wcond = [\,],$

$bsoln = [partsol = [1, 0, 0], basis = \{[-1, 0, 1], [-1, 1, 0]\}]]$

$[eqzro = [a^2 - ab - a + b^2 - b + 1], neqzro = [a - b^2], wcond = [\,],$

$bsoln = [partsol = [\frac{b - a}{b^2 - a}, 0, \frac{b - 1}{b^2 - a}], basis = \{[\frac{-b + a^2}{b^2 - a}, 1, \frac{-ab + 1}{b^2 - a}]\}]]$

$[eqzro = [a^2 - ab - a + b^2 - b + 1], neqzro = [ab - 1], wcond = [\,],$

$bsoln = [partsol = [\frac{b - 1}{ab - 1}, 0, \frac{a - 1}{ab - 1}], basis = \{[\frac{-b^2 + a}{ab - 1}, 1, \frac{b - a^2}{ab - 1}]\}]]$

$$[eqzro = [\,], neqzro = [a + b + 1, a^2 - ab - a + b^2 - b + 1], wcond = [\,],$$

$$bsoln = [partsol = [\frac{1}{b+a+1}, \frac{1}{b+a+1}, \frac{1}{b+a+1}], basis = \{\}]]]$$

Comparing this result with that given by Gardini *et al.* (p. 456 and Table 1, Column 1, p. 460), we see that our algorithm gives a more complete description. Indeed, only three points were given for the first and third regimes, whereas for the first regime, the solution set is actually a hyperplane (in 3-dimensional space) and for the third regime, each solution set is a line. Moreover, the condition for the equilibrium point in the last regime in that table was incomplete: for example, if $a = 1$ and $b = 1$ then $a + b \neq -1$, but since the second non-zero condition is not satisfied, this case belongs to the first regime. We note that the second regime is actually identical with the third by Proposition 6.2 since

$$\mathrm{Rad}(a^2 - ab - a + b^2 - b + 1, a - b^2) = \mathrm{Rad}(a^2 - ab - a + b^2 - b + 1, ab - 1).$$

This was not detected by the program because we have not used radical membership testing (when applying Corollary 6.3). Using Routine 4M instead of Routine 4I removes this redundancy. □

EXAMPLE 8.3. In this example, we solve a 5×4 PSLE with arbitrary right-hand side. Note that the output consists of three special solution functions, two with C-rank $= 4$ and one with C-rank $= 3$. In the former case, there are five 4×4 submatrices with non-zero determinants, but only two are irredundant. In the latter case, one of the 3×3 submatrices has a non-zero constant determinant, and hence the corresponding special regime covers all others with the same C-rank.

$r := 5; n := 4;$
Coeff : M *polring* := zero(r, n);
Coeff(0) := [0, 1, 0, 2] :: V *polring*;
Coeff(1) := [a, b, c, 1] :: V *polring*;
Coeff(2) := [2, 0, 1, d] :: V *polring*;
Coeff(3) := [0, 0, $-1, a$] :: V *polring*;
Coeff(4) := [1, 0, 0, 1] :: V *polring*;
Coeff

$$\begin{bmatrix} 0 & 1 & 0 & 2 \\ a & b & c & 1 \\ 2 & 0 & 1 & d \\ 0 & 0 & -1 & a \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

Avector: = [w_1, w_2, w_3, w_4, w_5];
psolve(*Coeff*, *Avector*)

$$[[eqzro = [a + d - 2, b + \frac{1}{2}cd - c - \frac{1}{2}d + \frac{1}{2}], neqzro = [\,],$$

$$wcond = [2w_5 - w_4 - w_3, (d + 2c - 2)w_4 + (d - 2)w_3 + 2w_2 + ((c - 1)d - 2c + 1)w_1],$$

$$bsoln = [partsol = [\frac{w_4 + w_3}{2}, w_1, -w_4, 0], basis = \{[-1, -2, -d+2, 1]\}]]$$

$$[eqzro = [\,], neqzro = [a + 2b + cd - 2c - 1],$$

$$wcond = [(-ad + 2ac - 4b - a^2 + 2)w_5 + (cd - 2c + 2b + a - 1)w_4$$
$$+ (-ac + 2b + a - 1)w_3 + (d + a - 2)w_2 + (-bd + (-a + 2)b)w_1],$$

$$bsoln = [partsol = [\frac{(cd + 2b - 1)w_5 - cw_3 + w_2 - bw_1}{cd - 2c + 2b + a - 1},$$

$$\frac{(4c - 2a)w_5 - 2cw_3 + 2w_2 + (cd - 2c + a - 1)w_1}{cd - 2c + 2b + a - 1},$$

$$\frac{(-ad - 4b + 2)w_5 + (2b + a - 1)w_3 + (d - 2)w_2 + (-bd + 2b)w_1}{cd - 2c + 2b + a - 1}$$

$$\frac{(-2c + a)w_5 + cw_3 - w_2 + bw_1}{cd - 2c + 2b + a - 1}],$$

$$basis = \{\}]],$$

$$[eqzro = [\,], neqzro = [a + d - 2],$$

$$wcond = [(-ad + 2ac - 4b - a^2 + 2)w_5 + (cd - 2c + 2b + a - 1)w_4$$
$$+ (-ac + 2b + a - 1)w_3 + (d + a - 2)w_2 + (-bd + (-a + 2)b)w_1],$$

$$bsoln = [partsol = [\frac{(d + a)w_5 - w_4 - w_3}{d + a - 2}, \frac{4w_5 - 2w_4 - 2w_3 + (d + a - 2)w_1}{d + a - 2},$$

$$\frac{-2aw_5 + (-d + 2)w_4 + aw_3}{d + a - 2}, \frac{-2w_5 + w_4 + w_3}{d + a - 2},]$$

$$basis = \{\}]]] \quad \square$$

EXAMPLE 8.4. We now modify the right hand side of Example 8.3 so that it involves the parameters a, b, c, and d. Notice that since w is the empty list (there are no new right-hand side parameters), $wcond = [\,]$ for all special solution functions. Note also that the first regime is finite, and the solution functions are expressed in terms of d alone.

```
u1, u2, u3, u4, u5 : polring;
u1 := a - b;
u2 := a + c;
u3 := 0;
u4 := a + b;
u5 := a - c + d;
Avector := [u1, u2, u3, u4, u5]
psolve(Coeff, Avector)
```

$$[[eqzro = [a + d - 2, b + \frac{2}{11}d^3 - \frac{9}{11}d - \frac{15}{11}, c - \frac{1}{11}d^3 - \frac{1}{11}d - \frac{7}{22},$$

$$d^4 - 6d^3 + d^2 + \frac{17}{2}d + 34],$$

$$neqzro = [\,],\ wcond = [\,],$$

$$bsoln = [partsol = [\frac{-2d^3-2d+37}{22}, \frac{2d^3-20d+7}{11}, \frac{2d^3+2d-37}{11}, 0\,],$$

$$basis = \{[-1,-2,-d+2,1]\}\,]\,],$$

$$[eqzro = [a^3+a^2b-3a^2c+2a^2d-2a^2-ab^2+abd-ab+2ac^2-4acd$$

$$+ac+ad^2-ad+a-b^2d-bcd-2bc+4bd+b-cd+4c-2d],$$

$$neqzro = [a+2b+cd-2c-1],\ wcond = [\,],$$

$$bsoln = [partsol = [\frac{cd^2+(-c^2+ac+2b-1)d+(-2b+2)c+b^2+ab}{cd-2c+2b+a-1},$$

$$\frac{((-b+a+4)c-2a)d-4c^2+(2b+4a+2)c+(-a+1)b-a^2+a}{cd-2c+2b+a-1},$$

$$\frac{-ad^2+((a+1)c+b^2+(-a-4)b-a^2+a+2)d+(4b-4)c-2b^2-2ab}{cd-2c+2b+a-1},$$

$$\frac{(-2c+a)d+2c^2+(-3a-1)c-b^2+ab+a^2-a}{cd-2c+2b+a-1}],$$

$$basis = \{\}\,]\,],$$

$$[eqzro = [a^3+a^2b-3a^2c+2a^2d-2a^2-ab^2+abd-ab+2ac^2-4acd$$

$$+ac+ad^2-ad+a-b^2d-bcd-2bc+4bd+b-cd+4c-2d],$$

$$neqzro = [a+d-2],\ wcond = [\,],$$

$$bsoln = [partsoln = [\frac{d^2+(-c+2a)d-ac-b+a^2-a}{d+a-2}, \frac{(-b+a+4)d-4c-ab+a^2}{d+a-2},$$

$$\frac{(-b-3a)d+2ac+2b-2a^2+2a}{d+a-2}, \frac{-2d+2c+b-a}{d+a-2}],$$

$$basis = \{\}\,]\,]\,]\ \square$$

There are variations of *psolve* that allow the user to selectively choose a C-rank and just solve for those special regimes with that C-rank. This feature can be quite handy when there are too many special solution functions and one may run out of memory. As implemented, the PLEQN package computes a small set of solution functions, which may be further processed. For example, if the regime is zero-dimensional (so that there are only a finite number of choices for the parametric values), one can actually solve the regime equations completely. The regimes can also be decomposed into irreducible components via primary decomposition or factorization, in which case, the solution functions may be simplified further on each component.

9. Comparison with Gaussian elimination

In this section, we return to analyze the Gaussian elimination method as applied to PSLE. Our aim is to prove a worst case complexity bound for the method, and to show that it is a lot larger than even the bound in Theorem 4.1. This complexity, in a sense, measures the number of distinct ways the Gaussian elimination may be executed when applied to all possible linear systems, and should be of independent interest, for example, as a starting point for an average-complexity theory.

By a *generic* parametric linear system, we mean a system L where $x = (x_{ij})_{1 \le i \le r, 1 \le j \le n}$, $w = (w_1, \ldots, w_r)$, $C_{ij}(x) = x_{ij}$, and $A_j(x,w) = w_j$. Of the three types of elementary row tranformations used in any Gaussian elimination scheme, we have to be particularly careful with the one that multiplies (or divides) a row by a "non-zero" entry $g(x)$. For PSLE, each such tranformation must be considered a pivoting step, and the process must branch unless $g(x)$ is a constant. Let G denote the binary tree corresponding to the Gaussian elimination algorithm: each non-zero, non-constant pivot element is represented by a node, with a left branch (*specifying* the element to be non-zero) and a right branch (*specifying* the element to be zero); a branch either leads to a node which is the next pivot in sequence or to a leaf which represents a decision that the system is consistent or not consistent. The (worst case) *complexity* of the Gaussian elimination algorithm as applied to PSLE is defined as the number of leaves of G. It measures the number of different paths that must be walked through while applying Gaussian elimination. In terms of solving a PSLE, this means the number of actual regimes (possibly empty) or systems (over $F(x, w)$) that must be tested for non-emptyness, consistency, and solved by back-substitution. In terms of solving a (non-parametric) linear system, this is the number of all possible ways the Gaussian elimination algorithm may be executed when applied to *all* possible linear systems of the given size. For the convenience of the derivation, we shall measure the size of a linear system by the size of the *augmented* matrix. Thus in what follows, a generic parametric linear system is given by an $r \times n$ augmented matrix x of indeterminates.

THEOREM 9.1. *Let L be a generic parametric linear system with an $r \times n$ augmented matrix x, $r \ge 1$, $n \ge 2$. Let $\varphi(r,n)$ be the complexity in applying the Gaussian elimination algorithm on L. Then*

$$\varphi(r,n) = \sum_{i=0}^{\min(r,n)} \binom{r}{i} \binom{n}{i} i!. \tag{12}$$

PROOF. Clearly $\varphi(1,n) = n + 1$. We define $\varphi(r,1)$ to be $r + 1$, which is the number of paths of the Gaussian algorithm when the coefficient matrix is the zero matrix, and the

right-hand side is a vector of indeterminates, (r of these paths will lead to inconsistency, and the remaining path leads to the unique trivial solution). Now assume $r \geq 2$, $n \geq 2$, and consider the branch $x_{11} \neq 0$. In this branch, no more elements on the first row or first column will be a pivot. Any further branching will occur in a PSLE L' with an $(r-1) \times (n-1)$ augmented matrix. We claim that L' is also generic. Clearly, the augmented matrix of L' is given by $C' = (C_{ij}')_{2 \leq i \leq r,\, 2 \leq j \leq n}$, where

$$C_{ij}' = x_{ij} - x_{i1} \frac{x_{1j}}{x_{11}}.$$

Suppose L' is not generic, and the family C_{ij}' is algebraically dependent over U. Let $z = (z_{ij})_{2 \leq i \leq r, 2 \leq j \leq n}$ be an $(r-1) \times (n-1)$ matrix of indeterminates over U. Then there is a non-zero polynomial $p(z) \in U[z]$ such that $p(C') = 0$. Let d be the total degree of p and choose p such that d is minimal. Let $q(x) = x_{11}^d p(C')$. Then $q(x)$ is a polynomial in x which is identically zero. For any i and j, such that $2 \leq i \leq r$, $2 \leq j \leq n$, we have $0 = \partial q/\partial x_{ij} = x_{11}^d (\partial p/\partial z_{ij})(C')$. By our choice of p, $\partial p/\partial z_{ij} = 0$ for all i, j. Thus $p = 0$, a contradiction. This proves our claim.

It follows now that the branch $x_{11} \neq 0$ leads to $\varphi(r-1, n-1)$ paths. The branch $x_{11} = 0$ will lead to the branch $x_{21} \neq 0$ which yields also $\varphi(r-1, n-1)$ paths, or to the branch $x_{21} = 0$. Continuing this way, we come to the path starting with $x_{11} = 0, \ldots, x_{r1} = 0$ which clearly leads to $\varphi(r, n-1)$ paths. We have thus established that $\varphi(r,n)$ satisfies the recurrence

$$\varphi(r,n) = r\varphi(r-1, n-1) + \varphi(r, n-1) \quad (r \geq 2,\ n \geq 2). \tag{13}$$

The proof of the theorem can now be completed by a simple induction. Formula (12) holds when $r = n = 1$. Assuming (12) holds for all smaller values of $r + n$, we have, using (13),

$$\begin{aligned}
\varphi(r,n) &= r\varphi(r-1, n-1) + \varphi(r, n-1) \\
&= r \sum_{i=0} \binom{r-1}{i} \binom{n-1}{i} i! + \sum_{i=0} \binom{r}{i} \binom{n-1}{i} i! \\
&= \sum_{i=1} r \binom{r-1}{i-1} \binom{n-1}{i-1} (i-1)! + \sum_{i=1} \binom{r}{i} \binom{n-1}{i} i! + 1 \\
&= \sum_{i=1} \binom{r}{i} \binom{n-1}{i-1} i! + \sum_{i=1} \binom{r}{i} \binom{n-1}{i} i! + 1 \\
&= \sum_{i=0} \binom{r}{i} \binom{n}{i} i!,
\end{aligned}$$

where, in the last equality, we have made use of the binomial identity

$$\binom{n-1}{i-1} + \binom{n-1}{i} = \binom{n}{i}.$$

In the above computation, we have also made use of the common convention that $\binom{\mu}{\nu} = 0$ whenever $\mu < \nu$ so that all summations may be treated as summing to infinity. This completes the proof of Theorem 9.1. □

The above proof applies, with slight modification, to a fraction-free elimination algorithm such as Gauss-Bareiss reduction. While the proof of (12) shown is simple, its derivation from (13) is rather tricky and technical, and we omit it since it does not give any further insight to the problem. The author was not able to find a combinatorial proof of (12). The theorem shows that $\varphi(r,n) = \varphi(n,r)$, a fact that is not entirely obvious.

We are now in a position to compare the worst case behavior of the Gaussian elimination method and Algorithm 2. Returning to our notation earlier, let L be a PSLE as given by (1). The number of linear systems solved with the Gaussian elimination method is given by

$$\begin{aligned}\varphi(r, n+1) &= r\varphi(r-1, n) + \varphi(r, n) \\ &= r\sum_{i=0}\binom{r-1}{i}\binom{n}{i} i! + \sum_{i=0}\binom{r}{i}\binom{n}{i} i! \\ &= \sum_{i=0}\binom{r}{i}\binom{n}{i} i!\,(r-i+1).\end{aligned} \tag{14}$$

Comparing (14) with (8) we see that for each linear system (over polynomial rings) of order c solved by Theorem 4.1, the Gaussian elimination algorithm solves $c!\,(r-c+1)$ systems. If n is fixed, then asymptotically $\varphi(r, n+1)$ is r^n while λ is $r^n/n!$. A similar statement holds if r is fixed. Thus at least for generic PSLE, our method solves a lot less linear systems. For general PSLE, the Gaussian elimination method always yields disjoint regimes, and it does not take advantage of the algebraic relations among the coefficients of the linear system. The improvements given in Algorithm 2 explore such relationships to reduce the number of regimes. As mentioned at the beginning of the paper and illustrated by Example 2.1, the author believes that it is very difficult to reduce the many solution functions obtained by Gaussian elimination through merging. We also saw in §2 that it is no simple matter to simplify intermediate results during the branch and pivot process; more importantly, Theorem 9.1 indicates that this may be memory intensive for dense PSLE. Finally, consistency decision, non-emptyness of regimes, simplification of regime representations and solution functions all require some kind of Gröbner bases computations. The advantage of our algorithm is clear.

10. Conclusions

In this paper, we have given an efficient algorithm for solving parametric linear systems, which is equivalent to solving the linear system for all possible (and not just generic) choices of parametric values. We developed a theory that allows us to reduce the number of regimes and solution functions. We analyzed the Gaussian elimination

method and showed that in the worst case, our algorithm yields far less number of regimes. We discussed in great detail how our algorithm may be implemented, paying attention to the current state of the art of computer algebra systems. We explored different methods of implementations, and illustrated with examples the subtle effects of small changes in similar routines.

We found that there are still many unsolved problems, both for theory and for implementation. On the theory side, it is not known whether the Gaussian method can be improved for a special subclass of PSLE (for example, sparse ones). There is the difficult question of merging regimes, which is especially significant for pivot and branch methods. Our algorithm depends on properties of special solution functions, and the best results we can obtain are for semi-homogeneous systems for which we can guarantee irredundancy. We know very little about solution functions in general, and thus we have no results on minimality other than the simple sufficient condition in Corollary 4.3. On the practical side, the simplification of representations of quasi-algebraic sets and of functions defined on them are most important. We are not completely satisfied with the methods we presented for finding a minimum generating subset for radical ideals. Neither are we satisfied with the enumeration of subdeterminants by brute force.

Symbolic computation is now easily available and affordable. Many problems are solved symbolically, but often only for generic inputs. To solve symbolically for degenerate inputs leads invariably to the consideration of parametric systems. We believe many problems in parametric systems may be reduced to parametric linear systems. Future research on the problems above will no doubt lay the foundation for more general parametric systems.

Acknowledgement

The author thanks the referees for many constructive suggestions in the presentation. The reorganization of the paper, separating theory from implementation issues, was suggested by one referee, who also supplied some of the more recent references on computing primary decomposition and radical ideals. Example 2.1 and the discussion on Gauss-Bareiss reduction was inspired by the comments of the other referee. The author appreciates the generous support and assistance from many members of the Computer Algebra Group at Yorktown during the implementation of the algorithm.

References

Alonso, M. E., Mora, T., & Raimondo, M. (1990). Local decomposition algorithms. *Proc. AAECC 8*. To appear.

Arnon, D. S. & Mignotte, M. (1988). On Mechanical Quantifier elimination for elementary algebra and geometry. *J. Symbolic Computations* 5, 237 – 259.

Bareiss, E. H. (1968). Sylvester's identity and multistep integer-preserving Gaussian elimination. *Math. Comp.* **22**, 565 – 578.

Buchberger, B. & Loos, R. (1982). Algebraic simplification. In (Buchberger, B., Collins, G.E., & Loos, R. ed.) *Computer Algebra – Symbolic and Algebraic Computation*, 2nd ed. Springer-Verlag, 11 – 44.

Buchberger, B. (1985). Gröbner Bases: An algorithmic method in polynomial ideal theory. In (Bose, N.K., ed.) *Multidimensional Systems Theory*, D. Reidel Publishing Co., 184 – 232.

Buchberger, B. (1987). Applications of Gröbner bases in non-linear computational geometry. In *Trends in Computer Algebra.* Proceedings, International Symposium, Bad Neuenahr, May 19 – 21, 1987; *Lecture Notes in Computer Science*, **296**, 52 – 80.

Dube, T., Mishra, B., & Yap, C. K. (1986). Admissible orderings and bounds for Gröbner basis normal form algorithms. *Robotics Research Technical Report, Courant Institute of Mathematical Sciences.*

Eisenbud, D., Huneke, C., & Vasconcelos, W. (1989). Direct methods for primary decomposition. Preprint.

Faugère, J.C., Gianni, P., Lazard, D. and Mora, T. (1988). Efficient computation of zero-dimensional Gröbner bases by change of ordering. Preprint.

Gardini, L., Lupini, R., Mammana, C. & Messia, M. G. (1987). Bifurcations and transitions to chaos in three-dimensional Lotka-Volterra map. *SIAM J. Appl. Math.* **47**(3), 455 – 482.

Gebauer, R. & Möller, M. (1988). On an installation of Buchberger's algorithm. *J. Symbolic Computation*, **6**, 275 – 286.

Gianni, P., Trager, B. & Zacharias, G. (1988). Gröbner bases and primary decomposition of polynomial ideals. *J. Symbolic Computation* **6**, 149 – 167.

Gianni, P. & Mora, T. (1989). Algebraic solution of systems of polynomial equations using Gröbner bases. *Proc. AAECC 5, Lecture Notes in Computer Science*, **356**, 247 – 257.

Giusti, M. & Heintz, J. (1990). Un algorithme – disons rapide – pour la décomposition d'une variété algébrique en composantes irréducibles et équidimensionnelles. *Proc. MEGA '90.* To appear.

Goldman, L. (1987). Integrals of multinomial systems of ordinary differential equations. *J. of Pure and Applied Algebra*, **45**, 225 – 240.

Guiver, J.P. (1985). The equation $Ax = b$ over the ring $\mathbb{C}[z,w]$. In (Bose, N.K., ed.) *Multidimensional Systems Theory*, D. Reidel Publishing Co., 233 – 244.

Jenks, D. *et al.* (1984). *SCRATCHPAD II* An experimental computer algebra system, abbreviated primer and examples. *Mathematical Science Department, IBM Thomas J. Watson Research Center.*

Krick, T. & Logar, A. (1990a). Membership problem, representation problem and the computation of the radical for one-dimensional ideals. *Proc. MEGA '90*, Birkhauser. To appear.

Krick, T. & Logar, A. (1990b). An algorithm for the computation of the radical of an ideal in the ring of polynomials. Preprint.

Kobayashi, H., Moritsugu, S. & Hogan, R. W. (1989). On radical zero-dimensional ideals. *J. Symbolic Computation* **8**, 545 – 552.

Kredel, H. (1988). Admissible termorderings used in computer algebra systems. *ACM SIGSAM Bulletin* **22**(1), 28 – 31.

Lang, S. (1964). *Introduction to algebraic geometry*, 2nd ed. John Wiley Interscience Publishers, Inc.

Lazard, D. (1988). Solving zero-dimensional algebraic systems. Preprint.

Mishra, B. & Yap, C. (1989). Notes on Gröbner bases. *Information Sciences* **48**, 219 – 252.

Neff, C. A. (1989). Decomposing algebraic sets using Gröbner bases. *Computer Aided Geometric Design* **6**, 249 – 263.

Nijenhuis, A. & Wilf, H. (1978). *Combinatorial Algorithms*, 2nd ed., New York: Academic Press.

Robbiano, L. (1985). Term orderings on the polynomial ring. *Proc. EUROCAL 1985. Lecture Notes in Computer Science* **204**, 513 – 517.

Robbiano, L. & Mora, T. (1988). The Gröbner fan of an ideal. *J. Symbolic Computation* **6**, 183 – 208.

Robbiano, L. (1988). Computer and Commutative Algebra. In *Proc. AAECC 6, Lecture Notes in Computer Science* **357**, 31 – 34.

Savageau, M. A., Voit, E. O., & Irvine, D. H. (1987a). Biochemical systems theory and metabolic control theory: 1. Fundamental similarities and differences. *Mathematical Biosciences* **86**, 127 – 145.

Savageau, M. A., Voit, E. O., & Irvine, D. H. (1987b). Biochemical systems theory and metabolic control theory: 2. The role of summation and connectivity relationships. *Mathematical Biosciences* **86**, 147 – 169.

Seidenberg, A. (1974). Constructions in algebra. *Trans. Amer. Math. Soc.*, **197**, 273 – 313.

Sit, W. (1988). On Goldman's algorithm for solving first order multinomial autonomous systems. *Proc. AAECC 6, Lecture Notes in Computer Science* **357**, 386 – 395.

Sit, W. (1989). Some comments on term-ordering in Gröbner basis computations. *ACM SIGSAM Bulletin*, **23**(2), 34 – 38.

Stillman, M. (1986). Lecture notes from Short Course #1: Computational algebraic geometry, Part 2: Applications. Computers & Mathematics Conference, Stanford University, July 29 – August 2, 1986.

Weispfenning, V. (1987). Admissible orders and linear forms. *ACM SIGSAM Bulletin.* **21**(2), 16–18.

Weispfenning, V. (1990). Comprehensive Gröbner bases. *Technishe Berichte der Fakultät für Mathematik und Informatik Universität Passau.* MIP–9003.

Zariski, O., & Samuel, P. (1958). *Commutative Algebra,* Volumes I and II, Van Nostrand Co. Inc.